AD NUMBER

ADB286590

NEW LIMITATION CHANGE

TO

Approved for public release, distribution unlimited

FROM

Distribution authorized to U.S. Gov't. agencies only; Proprietary Information; Jan 2003. Other requests shall be referred to USAMRMC, Ft. detrick, MD 21702

AUTHORITY

USAMRMC ltr, dtd 28 July 2003

AD_____

Award Number: DAMD17-99-1-9034

TITLE: Ultrasound Imaging Initiative

PRINCIPAL INVESTIGATOR: Raj Shekhar, Ph.D.

CONTRACTING ORGANIZATION: The Cleveland Clinic Foundation
Cleveland, Ohio 44195

REPORT DATE: January 2003

TYPE OF REPORT: Final

PREPARED FOR: U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE U.S. GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

LIMITED RIGHTS LEGEND

Award Number: DAMD17-99-1-9034
Organization: The Cleveland Clinic Foundation

Those portions of the technical data contained in this report marked as limited rights data shall not, without the written permission of the above contractor, be (a) released or disclosed outside the government, (b) used by the Government for manufacture or, in the case of computer software documentation, for preparing the same or similar computer software, or (c) used by a party other than the Government, except that the Government may release or disclose technical data to persons outside the Government, or permit the use of technical data by such persons, if (i) such release, disclosure, or use is necessary for emergency repair or overhaul or (ii) is a release or disclosure of technical data (other than detailed manufacturing or process data) to, or use of such data by, a foreign government that is in the interest of the Government and is required for evaluational or informational purposes, provided in either case that such release, disclosure or use is made subject to a prohibition that the person to whom the data is released or disclosed may not further use, release or disclose such data, and the contractor or subcontractor or subcontractor asserting the restriction is notified of such release, disclosure or use. This legend, together with the indications of the portions of this data which are subject to such limitations, shall be included on any reproduction hereof which includes any part of the portions subject to such limitations.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

John Bauer _____

_____ _____

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 2003	Final (21 Dec 98 - 20 Dec 02)

4. TITLE AND SUBTITLE
Ultrasound Imaging Initiative

5. FUNDING NUMBERS
DAMD17-99-1-9034

6. AUTHOR(S)
Raj Shekhar, Ph.D.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The Cleveland Clinic Foundation
Cleveland, Ohio 44195

E-Mail: shekhar@bme.ri.ccf.org

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Original contains color plates: All DTIC reproductions will be in black and white.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution authorized to U.S. Government agencies only (proprietary information, Jan 03). Other requests for this document shall be referred to U.S. Army Medical Research and Materiel Command, 504 Scott Street, Fort Detrick, Maryland 21702-5012.

12b. DISTRIBUTION CODE

13. Abstract ***(Maximum 200 Words)*** *(abstract should contain no proprietary or confidential information)*

This objective of this project was to build a real-time 3D ultrasound imaging system for combat casualty care. The high frame rate necessary for real-time 3D imaging was obtained using a synthetic aperture beamforming technique. The technique uses a fraction of the transmit pulses required by a conventional imaging system and permits very rapid image acquisition with no degradation of image quality. A beamformer capable of generating 650 2D image per second was implemented using a network of high speed digital signal processors. These images were then formatted to form 20 volumes (3D images) per second using a transducer array mounted on a mechanically rocking assembly. The array also featured a new multi-layer transducer design for improved electrical matching and noise performance. Alongside the scanner development, an extensive set of software tools were developed to allow real-time visualization and provide easy and accurate analysis of the resulting 3D data. All components were integrated and a prototype system was successfully tested during the summer of 2002.

14. SUBJECT TERMS	15. NUMBER OF PAGES
ultrasound, 3-D imaging, image processing, synthetic aperture	92
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

TABLE OF CONTENTS

INTRODUCTION

Providing medical imaging in the field is difficult. The goal of our project was to develop a real-time three-dimensional (3D) ultrasound imaging system for remote casualty care. Two key components of the project were 1) the development of a scanner that could collect 3D data 40 to 80 times faster than current 3D imaging approaches, and 2) the development of a set of software tools for rapid image manipulation and analysis at a remote site. Our hypothesis was that real-time 3D image acquisition in combination with telemedicine would enable high-quality medical imaging in a combat zone. A medical corpsman would simply place a small probe on the patient and position the sample volume by viewing a real-time two-dimensional imaging of the anatomy. Once the probe was correctly placed, a three-dimensional data set would be recorded in real time (0.05 s for each 3D data set). Real time 3D data acquisition is critical to avoid motion artifacts that would otherwise degrade the image. Following data acquisition, the images would be transmitted to a central hospital where an expert clinician could "re-scan" the patient by looking at multiple 2D planes through the data sets or examine a computer reconstruction of the three-dimensional anatomy.

BODY

In earlier annual reports, we described our progress in each of the items outlined in the statement of work. Instead of duplicating reporting progress on intermediate milestones in this final report, we are reporting here on the final goals derived from the items outlined in the 2nd and 3rd years of the project. These milestones also correspond with the functional modules making up the real-time 3D ultrasound scanner and the 3D ultrasound imaging workstation.

Scanner/Array Development

- **Develop Transducer Array and Probe Head – COMPLETED**

A photograph of the ultrasound probe that was developed is shown in Figure 1. The probe contains a 64-element 3.5 MHz linear phased array. The array was fabricated from 3 layers of PZT with interleaved electrodes. The novel multi-layer transducer was developed for this project by Tetrad Corp. (Inglewood, CO) to improve the noise performance and sensitivity of the transducer. The array is mounted on a rocking mechanism that mechanically scans the array (+/- 30 degrees) in the elevation direction (perpendicular to the image plane). By mechanically rocking the array at 20 Hz during image formation, a 3D data set is collected in real time. More details on the transducer array are provided in [1] and in Appendix 1.

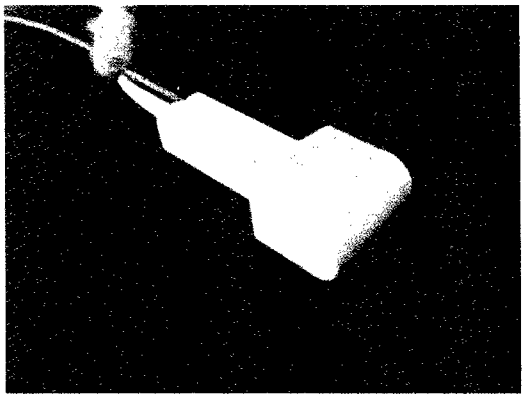


Figure 1 Ultrasound probe for the Phase I scanner.

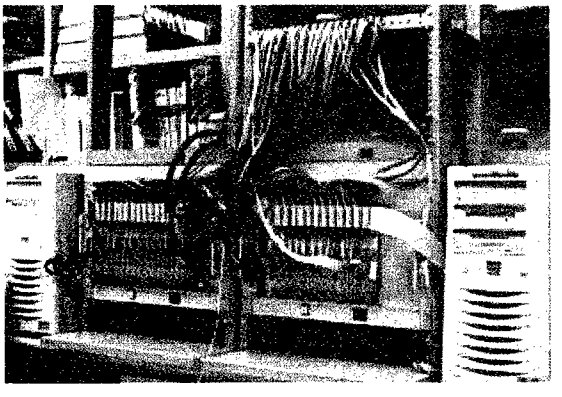


Figure 2 Synthetic aperture beamformer for the Phase I scanner.

- **Develop Beamformer – COMPLETED**

The heart of the 3D scanner is the beamformer shown in Figure 2. The beamformer was constructed using a parallel network of 32 DSP boards. Each board contains four TMS320C6201 DSPs, two channels of 12-bit, 42 MHz A/D, and two front-panel data ports all in a single VME slot. Each DSP processor can execute 16 instructions in parallel at 100 MHz, leading to a total processing speed of over 204 Giga operations per second. This provides enough computational power to beamform an entire 2D image in the time of a single transmit event. The beamformer, as currently implemented, is capable of generating 6.6 million points per second. This corresponds to a frame rate of approximately 650 2D frames per second for a 10,000-pixel

image. A detailed description of the beamformer is given in [2-6] and in Appendix 2. Following beamforming, the real-time 3D data set is transferred to a PC for real-time scan conversion and display.

- **System Integration & Image Preview – COMPLETED**

Image preview software is needed for the visualization of 3D ultrasound images at the site of the ultrasound scanner, as they are created. This capability is critical for ensuring that the probe makes good contact with the body and good quality images of the intended target organ are acquired. If not, the operator can use the instantaneous visual feedback to position and orient the probe correctly. A snapshot of a real-time 3D image of a string phantom collected using the integrated system is shown in Figure 3. The image is displayed with a dynamic range of 50 dB. The actual display consists of three intersecting planes (azimuthal [AZ] and elevation [EL] planes and a c-scan) through the volume, shown individually as well as together to illustrate the relative 3D spatial orientation among them (Figure 4). The displayed cross-sections are repainted as often as ultrasound volumes are generated. Controls are provided to pick the location of any of the three planes interactively. Additional controls allow saving the images for review by a clinical expert.

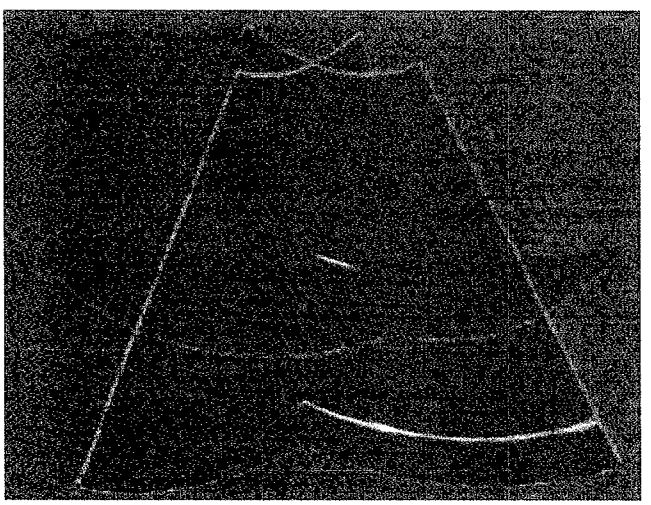


Figure 3 A snapshot of a real-time 3D image of the string phantom.

This visualization is based on a proprietary technique we have developed. The technique uses hardware-accelerated texture mapping capability that most commercially available graphics adaptors provide. Briefly, data samples corresponding to each image plane are extracted from the original 3D matrix representing a volume. The extracted data samples (a 2D matrix per plane) are then "texture mapped" on a number of quadrilaterals representing the ultrasound sector for AZ and EL planes and a curved rectangle for the c-scan. Polar-to-rectangular conversion or scan conversion of the displayed imaging planes takes place as part of the texture-mapping process. Our technique is both accurate and fast; it can scale to 70 Hz, well above the frame rate of a practical real-time 3D ultrasound scanner. The biggest advantage of our approach is that a task normally reserved for specialized hardware [7] is performed by consumer-variety graphics hardware without compromising image quality and speed. In addition to the real-time display, we developed a variety of tools for offline post-processing of the real-time volume data generated by the scanner.

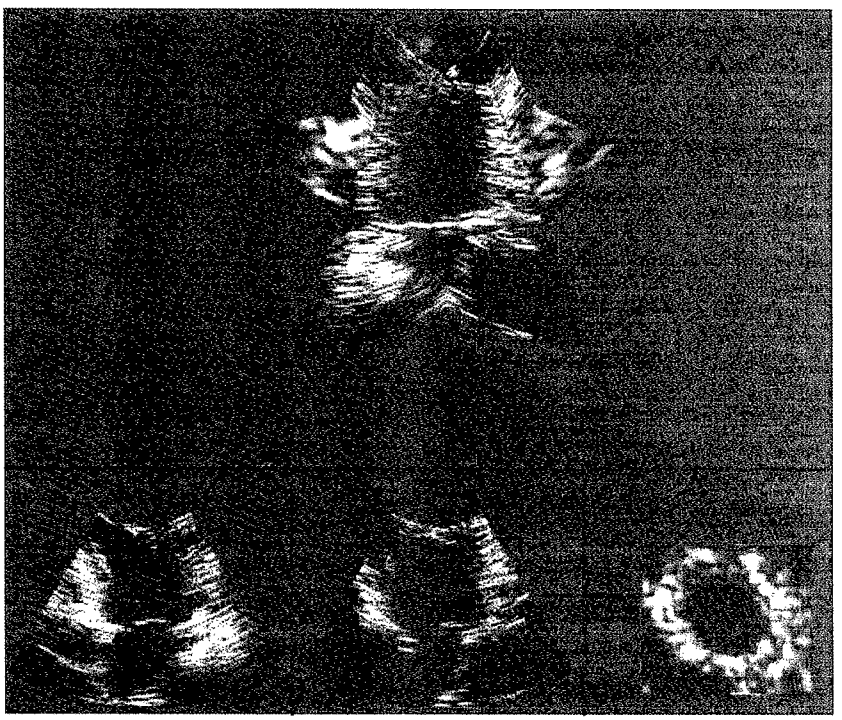


Figure 4 A snapshot of a scan-converting display. Shown in the bottom row are azimuthal, elevation and c-scan images. The top view shows all three images in 3D space.

User Interface & Image Processing Software

- **Develop Interactive 3D Visualization - COMPLETED**

This tool allows visualizing any of the infinitely many cross-sections through a volume, a capability called multi-planar reformatting (MPR). MPR is both dynamic and interactive - dynamic because we display time-varying data and interactive because simple mouse motion re-orients the image plane(s). In telemedicine, it is with this tool that a clinical expert, situated in a medical center, will virtually re-scan a patient. A snapshot of an interactive MPR session is shown in Figure 5.

MPR visualization is also based on texture-mapping hardware. One difference compared to image preview algorithm is that the MPR uses 3D texture mapping (instead of 2D), which provides greater flexibility to reconstruct even those scans, including oblique ones that were not acquired originally. Unlike online visualization, which is performed on native polar data due to time constraints, offline visualization works on 3D scan-converted Cartesian data. Because a typical 4-D image is almost always larger than the available texture memory, we have further developed custom volume subdivision and caching schemes to reduce the data requirement and optimize the performance. The method is described in detail in [8] and in a submitted paper (see Appendix 3).

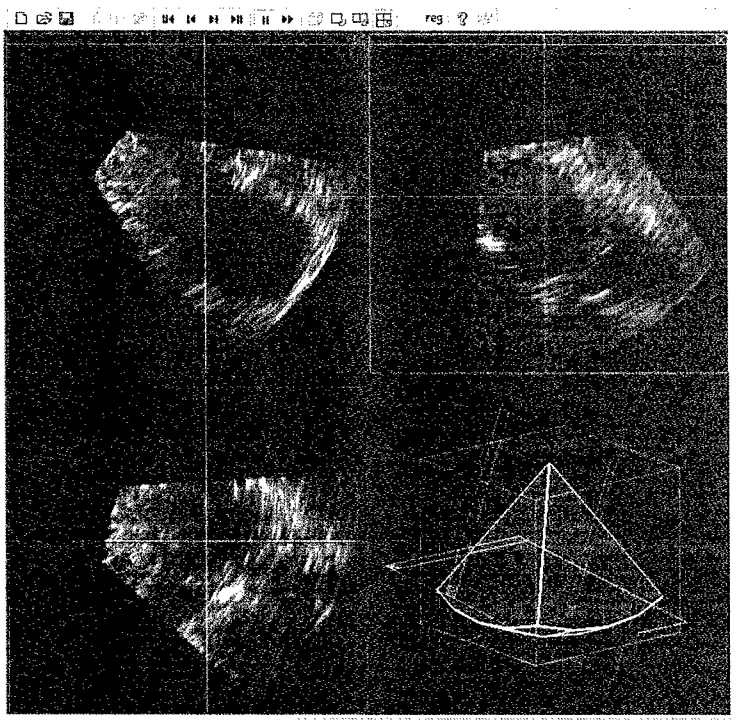


Figure 5 A static view showing three reformatted planes through a real-time 3D image of the heart. The bottom-right viewport contains a reference schematic that shows the orientation of the three planes with respect to the acquisition pyramid.

- **Develop Post Processing Algorithms - COMPLETED**

3D Image Segmentation

The 3D image segmentation we have developed is a generalization of the *Active Surface* segmentation technique [9,10] developed by our group. Significant phase I achievements include the development of a mesh library to represent closed 3D shapes and the determination of the expressions for internal and external forces that guide the morphing of a starting surface template to the desired organ shape. The internal force maintains the integrity of the mesh while it deforms under the influence of the external force derived from image gradients. The mesh library supports both contraction and expansion of meshes. Accuracy and stability was improved through the use of a generalized gradient vector field (GGVF) [11], whose effectiveness is well documented for defining the external force. The

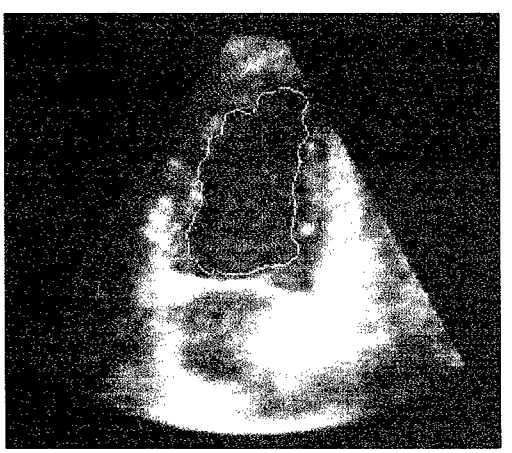


Figure 6 A 2D scan to show the contour of the endocardium following 3D segmentation.

GGVF helps increase the capture range of the mesh and permits convergence irrespective of whether the mesh needs contraction or expansion, locally or globally. The growth of an initial mesh to ultimately snap to the endocardium of the left ventricle (LV) for a diastolic frame is shown in Figure 6. By segmenting the endocardium in all phases, we have developed the capability to display the beating LV.

3D Image Registration

Image registration is the crucial first step when comparing serial images taken at different times and overlaying complementary images from multiple modalities. We have developed a mutual information-based method for registering ultrasound volumes and shown that the method is accurate, reliable and robust and applicable to rigid as well as nonrigid registration problems [12]. Furthermore, the method does not require placing external markers and is fully automatic as long as the initial misalignment is within the capture range. The method is applicable to a pair of single-modality (ultrasound + ultrasound) or multi-modality (ultrasound + nuclear medicine) images. Examples showing recovery of rigid and nonrigid misalignments using our method are shown in Figure 7. Two specific cardiac applications built upon this image registration technology are described in the "Cardiac Imaging Toolbox" subsection below.

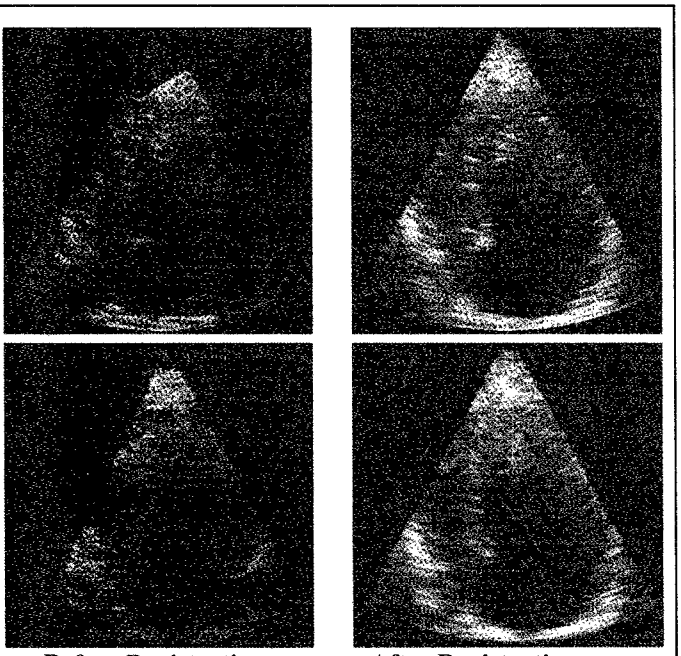


Figure 7 Recovery of rigid (top row) and nonrigid (bottom row) misalignment between a pair of ultrasound volumes.

Hardware Accelerator for Real-Time 3D Image Registration

Although the mutual information-based 3D image registration is automatic, accurate and versatile, its iterative nature makes it slow. Depending on the size of the image volumes, the degree of starting misalignment and the complexity of transformation (rigid vs. nonrigid), the execution time could reach 30 minutes. We have designed a custom hardware architecture that uses pipelining, parallel memory architecture and distributed computing to accelerate image registration [13,14]. We call this architecture fast automatic image registration (FAIR). The accuracy of the hardware implementation is the same as that of the software implementation. The expected speedup in the execution time, compared to a software implementation on an 800-MHz Pentium III microprocessor with a 133 MHz memory bus. A 29-fold faster execution is expected for 128^3-sized images when using two modules. In other words, if 200 iterations were needed before the algorithm converged, the registration would take approximately 24 seconds. Even faster execution with virtually no limit on performance can be achieved by increasing the number of modules.

- **Develop Toolbox For Detection Of Shrapnel – NOT COMPLETED**

Unavailability of ultrasound images showing shrapnel or trauma prevented the development of this toolbox. However, we plan to collect these images at a military hospital in the next phase and develop feature detection techniques similar to those developed for the breast imaging toolbox.

- **Develop Cardiac Imaging Toolbox – COMPLETED**

The first and foremost feature of the cardiac imaging toolbox is that it allows computing global function parameters. Using the above-mentioned 3D image segmentation, we can identify the endocardium of the LV over the entire cardiac cycle. In Figure 8, the computed left ventricular volume is plotted against time. Using this volume plot, we are able to compute end-diastolic volume, end-systolic volume, stroke volume and ejection fraction.

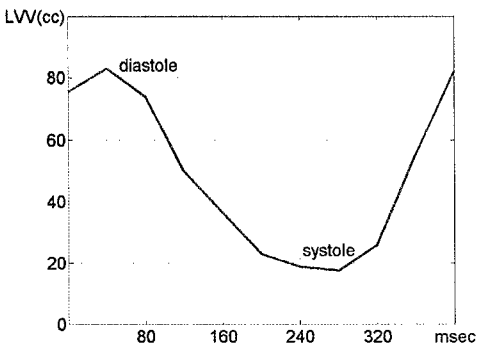


Figure 8 Volume of the left ventricular cavity over a cardiac cycle. The result was obtained following 3D segmentation of a real-time cardiac image.

The toolbox also features two novel applications – 3D stress echocardiography and multimodality cardiac fusion – enabled by real-time 3D ultrasound and 3D image registration technologies. Stress echocardiography is a common procedure in which heart motion in response to exercise (or other forms of stress) is compared with the motion of the resting heart. Partial data collection by conventional ultrasound and misaligned images are a common problem, which affect the accuracy of the procedure. We overcome these limitations by replacing conventional ultrasound with real-time 3D ultrasound in the existing stress protocol. Real-time 3D ultrasound shortens imaging time while providing complete data. As the next step, we correct any misalignment using the 3D registration technique and use the developed offline visualization to expose any anatomical section through the pre- and post-stress images [15].

While the ultrasound captures the structure and motion of heart, nuclear medicine imaging provides complementary perfusion information. We use the developed registration technology to spatially and temporally align and fuse 3D ultrasound and nuclear medicine cardiac image sequences. An example of this multimodality fusion is shown in Figure 9. This new procedure promises to improve the diagnostic power of detecting myocardial diseases.

Figure 9 Fusion of real-time 3D ultrasound (right) and gated nuclear medicine (left) images. The fused image is in the middle.

- **Breast Imaging Toolbox – COMPLETED**

Breast sonography is an important adjunct to diagnostic mammography and has primarily been used to distinguish between mammographically identified cystic and solid masses. It has not been used to differentiate between benign and malignant solid masses because of the apparent similarity of their sonographic appearance. We investigated the use of Haralick's texture features and posterior acoustic attenuation descriptors extracted from 2D breast ultrasound images for the characterization of breast masses as either cysts or benign or malignant solid masses. Separate models were constructed for distinguishing cysts from noncysts and benign solid masses from malignant lesions. Various candidate models were tested on 71 breast ultrasound images consisting of 24 cyst, 21 benign solid mass and 26 malignant solid mass cases confirmed by biopsy, and the best models were determined [16] and adapted in the breast imaging toolbox. Computerized analysis has the potential to increase the specificity of breast sonography and aid in better characterization of solid lesions.

KEY RESEARCH ACCOMPLISHMENTS

- Successful testing of fully integrated scanner prototype in the summer of 2002
- Successful development of a 64-element, 3-layer PZT transducer array mounted on a mechanically rocking assembly
- Successful development of a synthetic aperture, 64-channel parallel beamformer using a network of 32 digital signal processing boards
- Successful development of transmit and receive electronics for the synthetic aperture beamforming-based 3D ultrasound scanner
- Successful development of a real-time 3D scan converting display
- Successful development of interactive and dynamic multi-planar reformatting for virtual patient re-scanning
- Successful development of 3D image segmentation and registration algorithms
- Successful development of cardiac and breast imaging toolboxes

REPORTABLE OUTCOMES

Manuscripts, abstracts, presentations

[1] C.R. Hazard and G.R. Lockwood, "Theoretical assessment of a synthetic aperture beamformer for real-time 3D imaging," *IEEE Trans. Ultrason., Ferroelect., Freq. Contr.*, vol. 46, pp. 972-979, 1999.

[2] C.R. Hazard and G.R. Lockwood, "Effects of motion on a synthetic aperture beamformer for real-time 3D ultrasound," *Proc. 1999 IEEE Ultrason. Symp.*, Lake Tahoe, Nevada, vol.2, p.1221-1224, 1999.

[3] V. Zagrodsky, R. Shekhar, and J.F. Cornhill, "Mutual information based registration of cardiac ultrasound volumes," *Proceedings of SPIE Medical Imaging 2000*, San Diego, California, vol.3979, p.1605-1614, 2000.

[4] C.R. Hazard and G.R. Lockwood, "Developing a High Speed Beamformer Using the TMS320C6201 Digital Signal Processor," *Proc. 2000 IEEE Ultrasonics Symposium*, San Juan, Puerto Rico, vol.2, p.1755-1758, 2000.

[5] V. Zagrodsky, R. Shekhar, and J.F. Cornhill, "Multifunction extension of simplex optimization method for mutual information based registration of ultrasound volumes," *Proceedings of SPIE Medical Imaging 2001*, San Diego, California, vol.4322, p.508-515, 2001

[6] R. Shekhar and V. Zagrodsky, "Mutual information-based rigid and nonrigid registration of ultrasound volumes," IEEE Transactions on Medical Imaging, 21(1), p.9-22, 2002

[7] M. Inerfield, G.R. Lockwood, and S.L. Garverick, "A sigma-delta-based synthetic aperture beamformer for real-time 3D ultrasound," *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control,* 49(2), p 243-254, 2002.

[8] C.R. Hazard and G.R. Lockwood, "Real-time synthetic aperture beamforming: practical issues for hardware implementation", *Proceedings of the IEEE Ultrasonics Symposium,* v.2, p1513-1516, 2001.

[9] J.A. Brown and G.R. Lockwood, "A low-cost, high-performance pulse generator for ultrasound imaging," *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 49, pp. 848-851, 2002.

[10] R. Shekhar and V. Zagrodsky "Interactive visualization of four-dimensional ultrasound data," *Medicine Meets Virtual Reality 10/02*, p 485-487, 2002.

[11] R. Sivaramakrishna, K. A. Powell, M. L. Lieber, W. A. Chilcote and R. Shekhar, "Texture analysis of lesions in breast ultrasound images," *Computerized Medical Imaging and Graphics*, vol.26, p.303-307, 2002.

[12] M. J. Zipparo and C.G. Oakly, "Advanced transducer materials for ultrasonic imaging probes," *Proceedings of Ultrasonics Symposium*, 2002, In Press.

[13] R. Shekhar and V. Zagrodsky, "CineMPR: interactive multi-planar reformatting of four-dimensional cardiac data using texture mapping hardware," *IEEE Tranactions on Information Technology in Biomedicine,* Submitted.

[14] C.R. Castro Pareja, J.M. Jagadeesh and R. Shekhar R, "FAIR: A hardware architecture for real-time 3D image registration," *IEEE Transactions on Image Processing*, Submitted.

Degrees obtained

- Kurtis A. Swinehart, Masters of Science, Case Western Reserve University, 1999
  Thesis Title: Signal Conditioning for Real-Time Three-Dimensional Echocardiography

- Christopher R. Hazard, Doctor of Philosophy, The Ohio State University, 2001
  Dissertation Title: Real-time three-dimensional ultrasound imaging using synthetic aperture imaging

- Michael Inerfield, Doctor of Philosophy, Case Western Reserve University, 2001
  Dissertation Title: Sigma-delta modulation in real-time three-dimensional sparse synthetic aperture ultrasound imaging systems

- Carlos R. Castro Pareja, Masters of Science, The Ohio State University, 2002
  Thesis Title: An architecture for real-time image registration

Funding obtained

- Development of quantitative real-time 3D stress echocardiography (PI: Raj Shekhar)
  Funding Agency: The Whitaker Foundation
  Duration and Total Costs: 3 years (Sept 01-Aug 04) and $239,939

- Development of Real-time Three-Dimensional Image Registration (PI: Raj Shekhar)
  Funding Agency: National Medical Technology TestBed (NMTB)
  Duration and Total Costs: 1 year and $280,953

Funding applied for

- Improving coronary artery disease diagnosis through multimodality cardiac image fusion (PI: Raj Shekhar)
  Funding Agency: NSF

CONCLUSIONS

We have successfully developed a prototype of a high-speed ultrasound imaging scanner and demonstrated the feasibility of imaging an entire volume in the time normally required for a single 2D image. This scanner is based on synthetic aperture beamforming, which permits 40 to 80 times faster acquisition than a conventional ultrasound scanner.

The scanner that we developed has many novel features, including a new beamformer design, a new transducer array based on a multi-layer transducer technology, a fast scan conversion technique and several image analysis and measurement tools. The project was an ambitious one and the success of the scanner depended on the simultaneous application of many new technologies. During the summer of 2002, a working prototype was successfully tested. A detailed description of the design and performance of the prototype scanner is described in a number of publications that were listed under reportable outcomes heading.

Successful completion of this project has moved us closer to the goal of providing ultrasound imaging for forward echelon combat casualty care by using real-time 3D imaging in combination with telemedicine and a set of image analysis tools. This technology promised to prove equally effective in civilian emergency and non-emergency care.

REFERENCES

1. M. J. Zipparo and C.G. Oakly, "Advanced transducer materials for ultrasonic imaging probes," *2002 Ultrasonics Symposium Proceedings*, In Press.
2. C.R. Hazard and G.R. Lockwood, "Theoretical assessment of a synthetic aperture beamformer for real-time 3D imaging," *1998 Ultrasonics Symposium Proceedings*, IEEE Catalog No.#98CH36102, pp. 1865-1868, 1998.
3. C.R. Hazard and G.R. Lockwood, "Theoretical assessment of a synthetic beamformer for real-time 3D imaging," *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 46, pp. 972-980, 1999.
4. C.R. Hazard and G.R. Lockwood, "Developing a high speed beamformer using the TMS320C6201 digital signal processor," *2000 Ultrasonics Symposium Proceedings*, IEEE Catalog.#00CH37121, pp. 1755-1758, 2000
5. C.R. Hazard, "Real-Time Three-Dimensional Ultasound Imaging using Synthetic Aperture Beamforming," Ph.D. Thesis, The Ohio State University, June 2001.
6. C.R. Hazard and G.R. Lockwood, "Real-time synthetic aperture beamforming: practical issues for hardware implementation," *2001 Ultrasonics Symposium Proceedings*, IEEE Catalog.# 01CH37263, pp. 1513-1516, 2001.
7. C. Basoglu, Y. Kim and V. Chalana, "A real-time scan conversion algorithm on commercially available microprocessors," *Ultrasonic Imaging*, vol. 18, pp. 241-260, 1996.
8. R. Shekhar and V. Zagrodsky, "Interactive visualization of four-dimensional ultrasound data," *Medicine Meets Virtual Reality 10/02,* pp. 485-487, 2002.
9. R. Shekhar, R.M. Cothren, D.G. Vince, S. Chandra, J.D. Thomas and J.F. Cornhil, "Three dimensional segmentation of luminal and adventitial borders in serial intravascular ultrasound images," *Computerized Medical Imaging and Graphics*, vol. 23, pp. 299-309, 1999.
10. J.D. Klingensmith, R. Shekhar and D.G. Vince, "Evaluation of three-dimensional segmentation algorithms for the identification of luminal and medial-adventitial borders in intravascular ultrasound images," *IEEE Transactions on Medical Imaging*, vol. 19, pp. 996-1011, 2000.
11. C. Xu and J.L. Prince, "Generalized gradient vector flow external forces for active contours," *Signal P*rocessing, vol. 71, pp.131-139, 1998.
12. R. Shekhar and V. Zagrodsky, "Mutual information-based rigid and nonrigid registration of ultrasound volumes," *IEEE Transactions on Medical Imaging*, vol. 21, pp. 9-22, 2002.
13. C.R. Castro Pareja, "An architecture for real-time image registration," M.S. Thesis, The Ohio State University, March 2002.
14. C.R. Castro Pareja, R. Shekhar and J.M. Jagadeesh, "Architecture for real-time 3D image registration," United States Patent Application, Filed April 2002.
15. R. Shekhar, V. Zagrodsky, M. Garcia and J.D. Thomas, "3D Stress Echocardiography: A novel application based on registration of real-time 3D ultrasound images," *Proceedings of CARS 2002*, pp. 873-878. 2002.
16. R. Sivaramakrishna, K.A. Powell, M.L. Lieber, W.A. Chilcote and R. Shekhar, "Texture analysis of lesions in breast ultrasound images." *Computerized Medical Imaging and Graphics*, vol. 26, pp. 303-307, 2002.

APPENDICES

Appendix 1: Preprint of the publication - M. J. Zipparo and C.G. Oakly, "Advanced transducer materials for ultrasonic imaging probes," *Proceedings of Ultrasonics Symposium*, 2002. (6 pages)

Appendix 2: Title page and chapter 3 of Christopher R. Hazard's Ph.D. Dissertation titled, "Real-time three-dimensional ultrasound imaging using synthetic aperture imaging." (44 pages)

Appendix 3: Preprint of the submitted manuscript: R. Shekhar and V. Zagrodsky, "Cine MPR: Interactive multi-planar reformatting of four-dimensional cardiac ultrasound using hardware-accelerated texture mapping," submitted to *IEEE Transactions on Information Technology in Biomedicine*. (24 pages)

Advanced Transducer Materials for Ultrasonic Imaging Probes

M. J. Zipparo and C. G. Oakley
Tetrad Corporation, Englewood, CO 80112

Abstract: This paper presents development results of advanced materials applied to ultrasound arrays. Included in this are multilayer ceramics fabricated using a bonding process, and single crystal materials in both single layer and multilayer forms. Examples of resonator properties measured for these materials are presented. Single crystal arrays modeled using the finite element method (FEM) are compared to KLM modeled and measured. Lastly, a theoretical analysis of the transmit efficiency of arrays fabricated with both "hard" and "soft" PZT piezoceramic materials is compared, including proposed PZT4 multilayers. These should be useful for building high efficiency high power arrays which do not sacrifice imaging performance.

I. Introduction

The piezoelectric material can be considered the heart of any ultrasound imaging system, as it is responsibe for transducing energy on both transmit and receive at least once for every scan line. Poor array performance can often not be compensated for by adjustment of other system components.

State of the art arrays are pushing the limits of traditional piezoceramic materials such as PZT5-H. An example is multirow arrays with adjustable elevation focus capabilities. The elevation aperture is broken into multiple independent elements, the individual area of which is considerably smaller than for a comparable 1-D array. The result is array elements with very low capacitance and very high impedance, factors which ultimately lead to reduced sensitivity. Two dimensional phased arrays push this to the limit with element capacitance often only a few pF, making it impractical to drive a coaxial cable with a capacitance of 50 to 100 pF/m.

Another example is phased arrays for harmonic imaging. These arrays necessarily have a very fine pitch to avoid grating lobes when receiving 2nd harmonic content, and thus they also suffer from low element capacitance. To exacerbate the problem, the bandwidth requirement for optimum performance is at least 80 to 90%, preferably more.

Lastly, internal losses in piezoceramics which are piezoelectrically "soft" is high enough to cause excessive heating of the array in some imaging modes such as color Doppler or ARFI [1] where multiple transmit cycles are employed. To limit heating, fewer excitation cycles must be employed. This has the deleterious effect of decreasing resolution and signal to noise in these applications. Arrays for applications where imaging and therapy are combined require even higher power capability. To date, these arrays have achieved efficiency by sacrificing imaging performance [2, 3].

This paper presents experimental results which compare a conventional "soft" PZT (PZT5-H) to that same material in multilayer form and to single crystals in both single and multilayer forms. FEM modeling of a single crystal array shows closer agreement to measured than what can be achieved with the KLM model. A theoretical analysis also compares array efficiency for arrays employing both PZT5-H and a "hard" PZT (PZT4). As expected, PZT4 with a modified array architecture can lead to arrays which tolerate high power levels. Simulations show that a proposed PZT4 multilayer maintains high power capability while making bandwidth comparable to conventional arrays. This will be important for arrays which guide therapy and monitor treatment in imaging-therapy applications.

II. Comparison of Ceramic and Single Crystal

Multilayer Resonator Measurements

Table 1 shows measured array element data for single crystal PMN-32%PT from TRS Ceramics, Inc. (State College, PA) and PZT5-H (CTS 3203HD), including 1-, 2-, and 3-layer air-loaded resonators. It can bee seen that in single layer form the crystal material has lower clamped dielectric constant and a significantly higher coupling. The slightly lower velocity of the crystal and the higher coupling result in similar element capacitance for array elements made from the two materials and operating at the same frequency.

Table 1 – Measured resonator properties for single layer and multilayer resonators of ceramic and single crystals.

Sample	tk	ε_{33}^S	k_{33}'	Z	vel
	(mm)			(MRayl)	(mm/μs)
Single crystal					
1-layer	0.5mm	631	0.88	29.6	3.65
	0.53mm	829	0.91	29.6	3.66
	0.52mm	868	0.89	23.6	3.38
2-layer	0.2mm	3341	0.75	26.3	3.17
	0.3mm	3406	0.81	27.4	3.22
	0.4mm	1627	0.75	26.2	3.38
	0.6mm	3195	0.83	28.3	3.31
	0.95mm	3379	0.87	26.7	3.22
3-layer	0.6mm	8099	0.79	23.1	2.85
Piezoceramic					
1-layer		1263	0.71	31.6	4.05
2-layer		4860	0.70	28.5	3.65
3-layer		8732	0.62	27.5	3.53

In multilayer form the crystals maintained from 85 to 98% of the coupling of bulk crystal. The lower percentage is lower than that typically achieved with piezoceramic, while the upper percentage, achieved with a lower frequency resonator, is comparable to the best percent achieved with piezoceramic. The average multilayer coupling achieved represents a significant improvement over ceramics. The measured dielectric constants are close to the factor of N^2 expected.

The increased variability in performance seen for the crystal multilayers is likely due to these units being initial prototypes fabricated using a process designed for piezoceramics, rather than an inherent limitation of crystal materials.

The increased dielectric constant and high coupling of crystal multilayers is important for array designs which require high capacitance for elements with very small area.

Figure 1 shows an SEM micrograph of the bond between two ceramic plates in a multilayer structure. The interface runs horizontally just below the center of the photo and is barely discernable. For reference, the average grain size for this material is in the 3 to 5 μm range. A thin bond layer (e.g. << 1 μm for a 3 MHz resonator) between the plates is essential to achieve high coupling from the multilayers.

Figure 2 shows the percentage of coupling achieved for a number of 3-layer ceramic array element resonators operating at f_s = 3.25 MHz. The rightmost seven data points represent recent multilayers with very thin bond layers. It can be seen that refinement of and experience with the fabrication process has resulted in improved multilayer coupling, approaching that of bulk ceramic material.

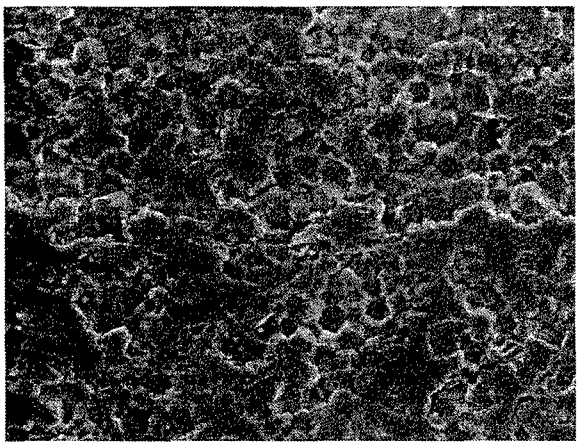


Figure 1 – SEM micrograph of bond interface between two ceramic plates in a multilayer structure.

Multilayers Applied to Multirow Arrays

Array prototype results for multilayer 1-D phased and linear arrays have been shown previously [4, 5]. Recent development work has resulted in multirow arrays where the elevation dimension has been divided into two to 10 independent elements, each with dimensions as small as 1.0 mm. Two dimensional phased arrays are even more limited by capacitance issues. The very small area of these elements results in a low capacitance high impedance element when single layer piezoceramics are employed. It is desirable to apply multilayer technology to these multirow arrays.

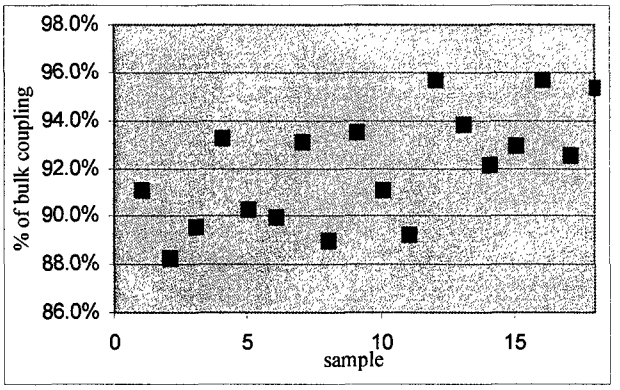


Figure 2 – Measured array element coupling relative to bulk array element coupling (k33' bulk = 0.705) for 3-layer ceramic resonators. The rightmost seven samples represent recent coupling results.

Multirow arrays with several elements in elevation make it impractical to use the current method developed for 1-D arrays to interconnect to the internal electrodes, and 2-D arrays are even more limited. The finite element program PZFlex (Weidlinger Associates) was used to investigate the effect of fringe fields on the coupling of multilayers with small elevation dimensions.

Depending on the methods employed to interconnect to internal electrodes, the wraparound area can extend over a portion or all of the element side. Figure 3 shows a plot of simulated array element coupling as a function of element elevation, including a curve for the case of full wrap around and the case of wrap around over only two thirds of the element side. In all the simulations the element thickness and width was kept constant.

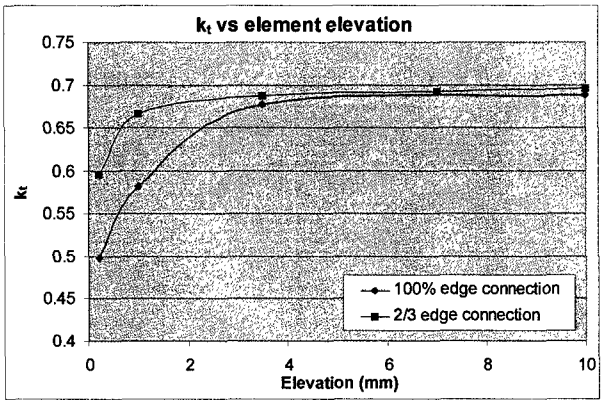


Figure 3 – Finite element modeling of the effective coupling of PZT5-H triple layers as a function of elevation length for full edge and 2/3 edge wraparound.

Regardless of the wraparound approach employed, elevations of 3.5 mm or more resulted in little degradation in coupling. For the partial wraparound the coupling remained at 97% of bulk for as little as 1 mm of elevation, then decreased fairly rapidly to 86% of bulk for 0.2 mm elevation, i.e. for a. 2-D phased array element. For the full wraparound the dropoff in coupling for elevations below 3.5 mm was steeper, reaching 83% of bulk for 1 mm elevation and 71% for 0.2 mm.

The reduction in coupling is due mainly to fringing electric fields in the regions where the electrode wraps around. It should be noted that these results are theoretical and do not factor in practical factors such as fabrication difficulties which are certain to further lower the coupling. While the partial wraparound solution is desirable from a performance standpoint, it may be more practical to build the full wraparound case. Reduced layer thickness would reduce the fringing field effect.

FEM Modeling of Crystal Arrays

The development of single crystal arrays has been hindered to a large degree by the difficulty and expense in obtaining high quality crystal material. Because of this it is desirable to use finite element modeling to hasten the development process. Until recently, modeling of crystals by FEA has been limited by the unavailability of an accurate matrix of properties for PMN-PT [6]. Now such a matrix is available [7].

Figure 4 shows comparison of measured and modeled array element responses for a single crystal linear array, including both the KLM model and PZFlex. It can be seen that both of the models agree fairly well over the main portion of the time waveform, with PZFlex agreeing better. It is the subtleties of the ringdown which determine such important factors as ripple in the spectrum. It is here that the PZFlex model predicts the response with more accuracy.

Efforts are underway to more closely match measured data for a variety of arrays. Ultimately it is hoped that FEA will prove valuable in developing designs with very wide bandwidth by limiting the amount of prototype fabrication which is necessary.

III. High Power, Efficient, Broadband Ultrasound Arrays

High power high efficiency arrays are often designed without consideration of bandwidth and pulse response. Conversely, imaging arrays are typically not designed to operate at high average power levels. This creates a difficult dilemma for dual-use imaging-therapy applications where high power and efficiency must be used concurrently with high quality imaging.

The factors which affect bandwidth and sensitivity have been discussed above, namely coupling and electrical impedance matching. Solutions to achieve very wide bandwidth exist in the form of multilayers and single crystals, and ultimately with multilayer single crystals.

These same technologies are applicable to solving the problem of how to achieve high efficiency and high power handling concurrently with wide bandwidth for high quality imaging. A study of loss mechanisms in conventional arrays on transmit is

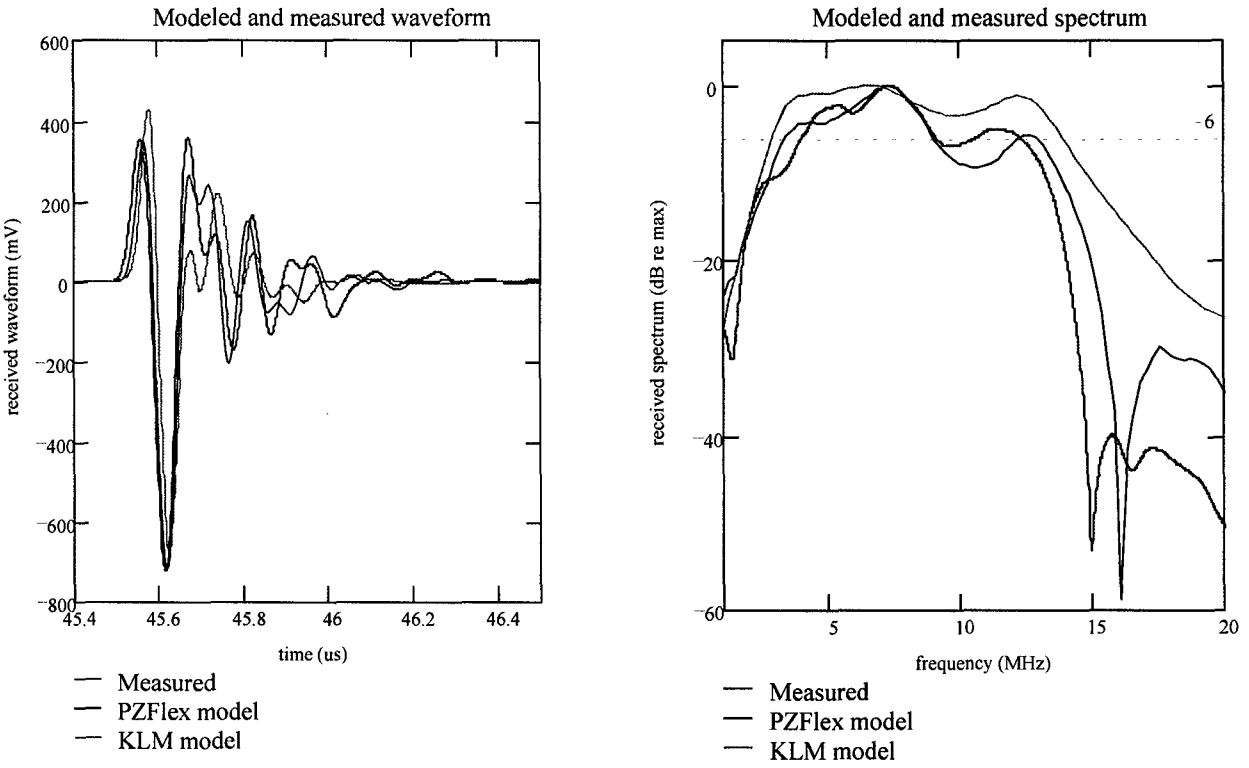


Figure 4 – Comparison of measured and simulated single crystal array waveform and spectrum using KLM and PZFlex models.

presented below. In addition the efficiency and bandwidth of improved designs using low loss PZT materials in single and multilayer form is shown.

Piezoceramic Properties at Low and High Average Power

Table 2 shows reported coefficients for various piezoceramic materials [8, 9]. It can be seen that the coupling coefficient and dielectric constants of PZT5-H are higher than the other ceramics. At high average power levels the Q_m for PZT5H decreases considerably [8]; the same should be true of PZT5-H. PZT4 and PZT8 are the only materials with both low dielectric and low mechanical loss terms. These terms change much less with increasing electric field than PZT5-H, with PZT8 changing less than PZT4 [8, p 85, Figure 12]. PZT4 has a slightly higher dielectric constant and coupling than PZT8.

Calculation of Transmission Efficiency and Round Trip Pulse Echo Response using the KLM Model

The coefficients of Table 2 were used in the KLM model to calculate the transmit efficiency and bandwidth for a variety of array configurations. These results are summarized in Table 3. Figure 5a shows the calculated loss characteristics of a conventional array using the low power characteristics of PZT5-H. It can be seen that the primary loss components are the backing and acoustic lens. The overall efficiency is 38%. When high power characteristics are employed the loss can be seen to be significantly higher (Figure 5b). It is clear why PZT5-H is unsuitable for applications which require transmission at high average power levels.

Employment of PZT4 can be seen to improve efficiency slightly. The primary benefit of using this material is that internal dissipation does not increase significantly when it is driven at high power levels. To further improve efficiency it is necessary to modify the transducer structure by not using a lens, by using a low impedance backing, or both. Unfortunately, these designs result in a tradeoff in performance in terms of bandwidth. This is due to both the reduced coupling and reduced dielectric constant of this material.

Table 2 - Reported material coefficients of various PZT compositions [8, *9]

material	$\varepsilon_{33}^T/\varepsilon_o$	$\varepsilon_{33}^S/\varepsilon_o$	k_{33}'	Q_m	Q_e	N_1	T_c	ρ	v_3^D	v	Z_a
						(MHz mm)	(deg C)	(g/cc)	(mm/μs)	(mm/μs)	(MRayl)
PZT4	1300	635	0.70	500	250	1650	328	7.5	4.60	4.08	30.6
PZT5-A	1700	830	0.71	75	50	1400	350 *	7.8	4.35	3.86	30.1
PZT5-H	3400	1470	0.75	65	50	1420	260 *	7.8	4.56	4.05	31.6
PZT7-A	425	235	0.66	600	60	1750	350	7.6	4.80	4.26	32.4
PZT8	1000	580	0.64	1000	250	1700	300	7.6	4.58	4.07	30.9

** Note: High average power drive properties are approximately Qm = 5 and Qe = 4.

Table 3 – Parameters used in the KLM and the resulting transmit efficiency and bandwidth.

		PZT5-H low power parameters	PZT5-H high power parameters	PZT4	PZT4 low impedance backing	PZT4 no lens	PZT4 low impedance backing and no lens	PZT4 triple layer, low impedance backing and no lens
Q_m lens		47	47	47	47	n/a	n/a	n/a
Q_m OML		17.1	17.1	17.1	17.1	17.1	17.1	17.1
Q_m IML		32.7	32.7	32.7	32.7	32.7	32.7	32.7
lens thickness	(mm)	0.381	0.381	0.381	0.381	0	0	0
$Z_{backing}$	(MRayl)	3.0	3.0	3.0	0.5	3.0	3.0	3.0
Q_m ceramic		65	5	250	250	250	250	250
tan(δ)		0.02	0.25	0.008	0.008	0.008	0.008	0.008
k_{33}'		0.705	0.705	0.66	0.66	0.66	0.66	0.6
$\varepsilon_{33}^S/\varepsilon_o$		1263	1263	635	635	635	635	5256
Transmit efficiency	(P_{out}/P_{in})	38	19	40	53	57	76	76
BW	(%)	76	87	59	37	59	37	85

Table 4 – Measured, reported, and calculated properties for single and multilayer piezoceramic resonators.

material	# layers		$\varepsilon_{33}^S/\varepsilon_o$	k_{33}'
PZT5-H	1-layer	measured	1263	0.71
PZT4	1-layer	reported	635	0.66
PZT8	1-layer	reported	580	0.61
PZT5-H	2-layer	measured	4860	0.65
PZT4	2-layer	calculated	2443	0.60
PZT8	2-layer	calculated	2231	0.55
PZT5-H	3-layer	measured	10500	0.65
PZT4	3-layer	calculated	5256	0.60
PZT8	3-layer	calculated	4800	0.55

III – High power multilayers

The degradation in performance which results from the lower dielectric constant and coupling of PZT4 can be largely offset by the use of a multilayer structure to improve electrical impedance matching. Table 4 shows the measured properties of PZT5-H ceramic array element resonators, including 1-, 2-, and 3-layer structures. Also shown are the reported coefficients of PZT4 and PZT8, and from these the calculated performance of proposed high power multilayer ceramics. The parameters for a triple layer PZT4 stack were input into the KLM model with no lens and a low impedance backing. The simulated efficiency was 76% with 85% bandwidth (Figure 5c).

IV. Conclusions

Arrays with very small element area are driving the need for multilayer technology in the low MHz range to improve element electrical matching. Fabrication of such multilayers must be done with a thin bond between the layers to maintain coupling. Multilayer crystals promise similar electrical impedance matching but significantly higher couplings, a factor which allows for very wide bandwidths to be achieved. Multilayers applied to multirow arrays must overcome the effect of fringing electrical fields which detract from the coupling.

High power arrays require the use of low loss piezoceramic materials. Historically incorporation of such materials into arrays designed for high transmit efficiency has resulted in poor bandwidth and subsequently poor imaging performance. Extrapolation of measured multilayer properties to similar designs with low loss PZT have shown that high bandwidth and high transmit efficiency do not have to be exclusive of each other. Development of high power multilayers should make possible the design and fabrication of arrays for dual use imaging-

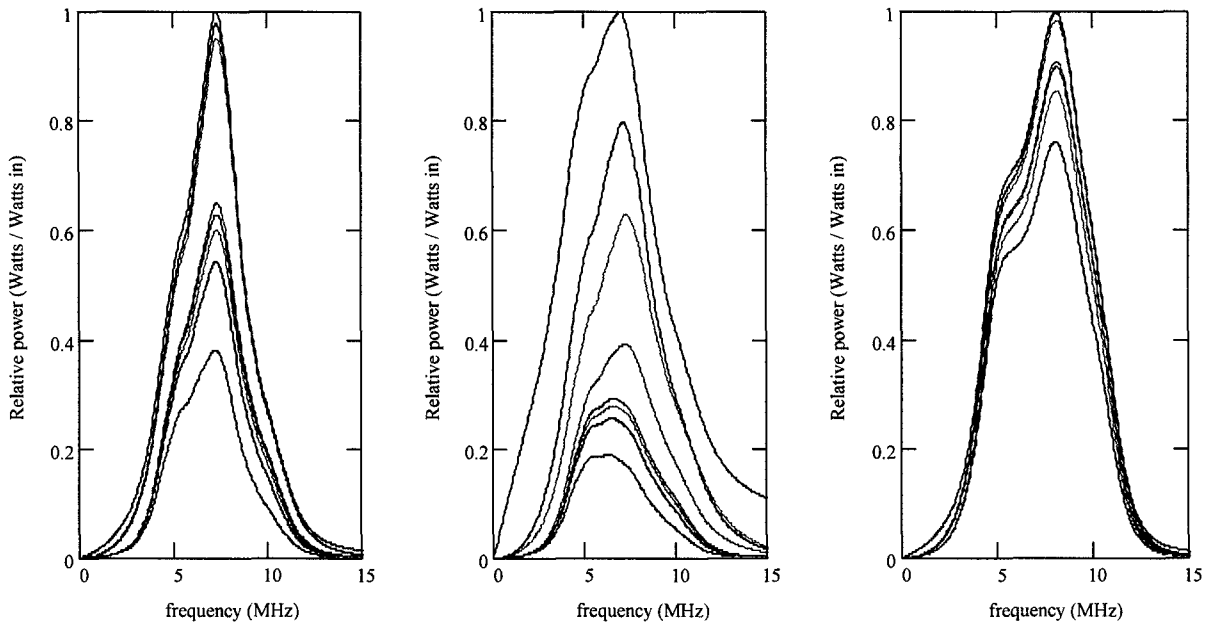


Figure 5 – Simulated power at KLM model terminals from the electrical input terminals to the front acoustic load. a) PZT5-H low power parameters with 3.0 MRayl backing and lens; b) PZT5-H high power parameters with 3.0 MRayl backing and lens; c) PZT4 with no lens and low impedance (0.5 MRayl) backing.

therapy applications where high efficiency and wide bandwidth are achieved simultaneously.

V. Acknowledgements

The authors would like to acknowledge the financial support of ONR and DARPA for a portion of this work.

VI. References

[1] K. Nightingale, et. al., "Acoustic radiation force impulse imaging: in vivo demonstration of clinical feasibility," *Ultrasound in Med. And Biol.*, vol. 28, no. 2, pp. 227 – 235, 2002.

[2] C. Simon, P. Van Baren, and E. Ebbini, "Combined imaging and therapy with piezocomposite phased arrays," *IEEE 1998 Ultrason. Symp. Proc.*, pp. 1555 – 1558.

[3] P. G. Barthe and M. H. Slayton, "Efficient wideband linear arrays for imaging and therapy," *IEEE 1999 Ultrason. Symp. Proc.*, pp. 1249 – 1252.

[4] M. Zipparo, C. Oakley, and M. He, "Multilayer ceramics and composites for ultrasonic imaging arrays," Presented at the 1999 IEEE Ultrasonics Symposium, Lake Tahoe, NV. Vol. 2, pp. 947-952.

[5] M. Zipparo and C. Oakley, "Medical imaging phased arrays using multilayer ceramics and composites," Presented at the 2000 IEEE Ultrasonics Symposium, San Juan, Puerto Rico, vol. 2, pp. 1169-1172.

[6] M.J. Zipparo and C.G. Oakley, "Finite element modeling of PZN-PT and PMN-PT single crystal materials," Presented at the 2001 IEEE Ultrasonics Symposium, Atlanta, GA.

[7] W. Cao, The Pennsylvania State University, personal communication.

[8] D. Berlincourt, "Piezoelectric crystals and ceramics," Ch. 2, pp. 100 - 107, in Ultrasonic Transducer Materials, O.E. Mattiat, Ed., New York, Plenum Press, 1971.

[9] http://www.ctscorp.com/pzt/pdf/PZT.pdf

REAL-TIME THREE-DIMENSIONAL ULTRASOUND IMAGING USING SYNTHETIC APERTURE BEAMFORMING

DISSERTATION

Presented in Partial Fulfillment of the Requirements for

the Degree Doctor of Philosophy in the Graduate

School of The Ohio State University

By

Christopher R. Hazard, M.S.

* * * * *

The Ohio State University
2001

Dissertation Committee:

Professor Geoffrey Lockwood, Adviser

Professor Mardi Hastings

Professor Randolph Moses

Approved by

_____

Advisor
Biomedical Engineering Graduate Program

CHAPTER 3

IMPLEMENTING A SYNTHETIC APERTURE BEAMFORMER USING DIGITAL SIGNAL PROCESSING HARDWARE

3.1 Introduction

This chapter describes the implementation of a prototype real-time 3D imaging system based on synthetic aperture beamforming. The beamformer must be able to calculate an entire two-dimensional image in the time a conventional beamformer would calculate a few lines, and calculate an entire volume in the time a conventional system would calculate a single image. Digital signal processors (DSP) provide an ideal platform for developing such a beamformer. DSPs provide the processing speed required for real-time beamforming, but are still easily adapted to different algorithms.

In the previous chapter, a beamforming architecture was introduced that accurately delayed signals using a linear interpolation algorithm, and summed values channel-to-channel using a pipelined network of DSPs. Here, that architecture is further investigated and the practical details of implementing it on hardware are considered. The system has been implemented using a fixed point DSP (TMS320C6201, Texas Instruments, Dallas, TX). An important limitation on the performance of a DSP based beamformer is the speed at which data can be moved between the multiple processors in the system. In fact, for a high-speed beamformer,

the input/output (I/O) performance is as important as the CPU speed. Algorithms directed at minimizing the I/O demands on the system have been developed and are presented. The issue of envelope detection and log compression of the beamformed signals using the fixed-point processors is discussed. Transfer of beamformed data from the DSP beamformer to a PC for scan conversion and display is also described for the system.

3.2 Overview of Pipeline Architecture

The beamformer consists of a network of digital signal processors. Each channel of the beamformer has an analog-to-digital converter (A/D) and two DSPs as well as the associated supporting hardware. Figure 3.1 shows the general layout of the DSP network. The beamforming sum is calculated by arranging the DSPs in a pipeline such that the input to a particular processor is the sum of all the prior channels and the output is the sum including the contribution from that particular channel. Pipelining can reduce the number of processors by eliminating the need for a hierarchical structure in which pairs of channels are summed by additional processors. However, a pipeline introduces two problems that the system must overcome. First there is a skew in the data that is being summed. Second, the data transfer between processors must occur simultaneously and at very high data rates.

3.2.1 Pipelining and Data Skew

Pipelining introduces a data skew because the N^{th} processor cannot add its contribution to the sum until all the processors up to and including the $(N-1)^{th}$

processor have calculated sums. This means that the N^{th} channel must either wait to calculate its contribution to a particular beamformed point or store the values until the previous channels have calculated the sums. Waiting to calculate the contribution is constrained by the amount of memory available to store the digitized signals that are required for the calculations. The system described here calculates the values and stores the results until they are ready to be summed. This minimizes the time between the transmit events for a particular 2D image. On the particular hardware used, it is more efficient to transfer data between processors in large blocks than to transfer individual data points. This means that large blocks of calculated values must be stored. The need to store data for the purpose of synchronizing the summing across the pipeline is referred to as data skew. The concept of data skew, as it is present in the system implemented, is illustrated in Figure 3.2.

Figure 3.2 shows several channels and several calculation cycles. To minimize the time between transmit events, calculations using the collected data must be carried out as soon after the collection as possible. Beamforming requires synchronized data collection on all channels and thus the calculations also occur nearly simultaneously for all channels. On Figure 3.2, all channels interpolate the same set of points during the same cycle. The elements of the vector ${}^{p}\mathbf{y}_{c}$ are interpolated values for the set of points p, being calculated at channel c. ${}^{p}\mathbf{S}_{c}$ is a vector composed of the sums for the set of points p, at channel c. Equation (1) shows explicitly that the sum at channel c depends on the sum at the previous channel, c-1.

$$ {}^{p}S_{c} = {}^{p}y_{c} + {}^{p}S_{c-1} \qquad (1) $$

As a result, the interpolated values must be stored until the sum propagates to each channel. Each channel needs enough memory to store c times the memory needed for one set of values.

Transferring data between processors is most efficient when done using the direct memory access controller (DMA). The central processing unit (CPU) must setup each access to external memory. The DMA reduces this overhead by allowing many data points to be transferred for a single setup. However, the DMA itself must be initialized for each block to be transferred and, therefore, data is most efficiently transferred in large blocks, which minimize the setup of the DMA and the external memory interface. The pipeline-induced data skew results in a trade-off between memory and I/O efficiency. The amount of memory in the current system limits the block size to 300 beamformed points.

3.2.2 Pipelining and Inter-Processor Communications

Another problem introduced by the pipelining architecture is the need for simultaneous transfer of data between processors. Digitized data comes into the system at 84 MB per second on each channel. Interpolated outputs are calculated at 160 MB per second on each channel. These large data rates preclude the use of standard bus architectures, since a bus would not allow the parallel transfer of data between channels. The solution the available DSP hardware provides is a 100 MHz, 32-bit first-in-first-out (FIFO) memory buffer and a 160 MB/s ribbon cable interface that directly connects channels. This allows the simultaneous transfer of sums between channels.

3.3 Particular Hardware Implementation

A linear interpolation synthetic aperture beamformer has been implemented using the 200-MHz, TMS320C6201 DSP. The system is based on a multi-processor board manufactured by Pentek, Inc. (Model 9137, Upper Saddle River, NJ). Each board provides four C6201 processors; two channels of 12-bit, 42-MHz A/D; and two Front Panel Data Ports (FPDP) in a single VME slot. The board consists of a 6U baseboard that houses the four C6201 processors and two 3U daughter boards in the single VME slot. One daughter board (Pentek Model 6211) provides two channels of A/D with a high speed FIFO interface between the A/D and the two associated processors. The second daughter board (Pentek Model 6226) provides two FPDP interfaces. The baseboard, along with the two daughter boards, represents two beamforming channels consisting of an A/D and two TI C6201s. One DSP is used to calculate delayed and apodized values using linear interpolation, while the second DSP sums the interpolated values as part of the previously described summing pipeline.

A simplified diagram of the architecture of the C6201 is shown in Figure 3.3. A few of the important features, relevant to programming the synthetic aperture beamformer, are described here. The processor runs at 200 MHz, resulting in a 5-nanosecond instruction cycle. The C6201 has two sets of functional units and register banks, labeled A side and B side on Figure 3.3. Each side of the processor has four functional units, which with some restrictions can be operated in parallel. The functional units include: an arithmetic logical unit (L), multiplier (M), shift register (S), and data unit (D). Each side has its own register file with 16 general-purpose

registers. The two sides communicate using a set of cross paths. Several types of memory are interfaced to the TI C6201 processor core. The internal program memory, 64kB of on-chip RAM, can be used either as a cache or as the primary storage for executed code. The beamformer stores all the executed code in program memory. For data storage, the most efficient memory is the 64 kB of internal data SRAM. The internal data SRAM is divided into two blocks with four banks in each. The block/bank architecture allows simultaneous accesses to different bank/blocks by both sides of the CPU and the DMA. Memories and communication structures located off the chip are connected to the processing core through the external memory interface (EMIF). The structures accessed through the EMIF include SB-SRAM, S-DRAM, global SRAM, and FIFOs. Since all of the external memories utilize the EMIF, only one external device can be accessed at a time. The C6201 also has a four-channel direct memory access controller. The DMA allows data transfers to occur in the background without use of the central processing unit[1].

Signals from the transducer are amplified, filtered and adjusted to the appropriate voltage range before being digitized at 41.75 MHz by the A/D. A clock distribution amplifier is used to synchronize data collection across the multiple A/Ds in the system. This distribution board synchronizes the system by transmitting the clock and enable signals to each board via an independent ribbon cable. Small and repeatable delay errors are present at each channel. Each of the 12-bit digital samples is stored as a two-byte half-word. Two A/D samples are then packaged into a single

32-bit word that is written into a FIFO at 84 MB/s. The data can be read in bursts from the FIFO by the C6201 at 400 MB/s. This allows the data to be read and processed in real-time.

A complete synthetic aperture image is formed using information collected from three separate transmit events. A desire to minimize the time between transmits for a particular image limits the number of transmits that the system can handle. Since there is a limited amount of the fast on chip SRAM, only a few transmits of data can be stored. The present implementation limits the system to three transmit events. Data for the first two transmits of an image are stored in the internal data memory on the TI C6201. The third transmit is temporarily stored in the FIFO itself, until processing of the first stored transmit is complete. If the time between transmits is not minimized, then more transmits can be collected by simply waiting until a transmit is completely processed before collecting the next transmit. However, collection of all three transmits in a time that is limited by the speed of sound and not the processing speed of the beamformer reduces the system's sensitivity to motion.

Figure 3.4 shows the relevant architecture for a single multiprocessor board from Pentek. The C6201s, connected to each A/D, calculate apodized and delayed values using the linear interpolation algorithm that was introduced in Chapter 2. In Figure 3.4, the interpolating C6201s, labeled C and D, read addresses and coefficients from the external SB-SRAM. The interpolators calculate the delayed values and store them temporarily in the internal data memory. This temporary storage helps offset the data skew introduced by the pipeline architecture. After interpolation the values are transferred to the second processor. This second C6201 provides additional skew

memory buffers and adds the values for that channel to the sum of the values from the previous channels in the pipeline. On Figure 3.4, the adders are labeled A and B. Each adder has three connections to the external memory interface of the C6201: a FIFO input from the interpolating C6201, a second FIFO input, and a FIFO output. For processor B, in Figure 3.4, the second FIFO input is from the 160-MB/s ribbon-cable interface that connects multiple boards. The ribbon-cable interface, referred to as a Front Panel Data Port (FPDP), provides the sum of the values from the previous channels on other boards. Processor B calculates a new sum by adding the values from processor D, which were temporarily stored in skew buffers. This new sum is then sent to processor A using the output FIFO for processor B. For procesor A, the second input FIFO is connected to processor B and contains the newly summed values. Processor A adds the values from processor C and sends the sum to the next board using the ribbon cable interface.

Two processors are associated with each beamforming channel in order to minimize the I/O required for any one processor. Figure 3.5 shows how adding a second processor reduces the I/O burden on any particular processor. In Figure 3.5a, a single processor is associated with each channel. This single processor would need to delay, apodize and sum the values. The external memory interface would be used to read digitized values from a FIFO connected to the A/D, to read addresses and coefficients for interpolation from the SB-SRAM, to read the partial sums from upstream in the pipeline, and to write the newly summed values to the FIFO connected to the next channel. Since all four of these transfers require the external memory interface, they cannot occur simultaneously. In Figure 3.5b, two processors are used

for each channel to reduce the I/O bottleneck for each processor individually. One processor does the interpolating and one is part of the summing pipeline. The interpolation processor has only three transfers which use the external memory interface: reading the A/D FIFO, reading the coefficients from SB-SRAM, and writing the interpolated values to a FIFO connected to the summing processor. Likewise, the summing processor also has three external memory accesses: reading the FIFO from the interpolating processor, reading the FIFO from the previous channel, and writing the sums to the FIFO connected to the next channel. Using two processors per channel reduces the number of external memory accesses for each processor. A practical reason for using two processors per channel is the fact that an A/D and FPDP interface cannot be connected to the same processing core with the available hardware. The addition of a second processor also increases the memory available for alleviating the data skew problem.

Figure 3.6 shows how the multiple boards are interfaced using the ribbon cable FPDP interfaces. In addition, the A/Ds from each channel are synchronized by distributing clock and start signals to each board via independently driven ribbon cables. Figure 3.7 shows the connections and signals involved with synchronizing the A/Ds and beamformer with the high-voltage pulser and the transducer's motion controller. As the transducer is rocked to sweep out a volume, a pulse is generated by the motion controller which signals that an image should be collected. This pulse from the motion controller is used to trigger the first transmit event for the image. Upon receiving the pulse from the motion controller, the synchronization distribution amplifier enables the A/Ds on all channels to collect data starting at the first clock

cycle after the pulse is received. The synchronization distribution amplifier also produces an output pulse, which is synchronized to the clock cycle on which the A/Ds start collecting data. This output pulse is used to trigger the high-voltage pulser that excites the transducer. Once the beamformer has finished collecting the data for the first transmit event in the image, the beamformer itself generates a pulse using a serial port on one of the C6201s. This pulse is also connected to the synchronization distribution amplifier. The pulse from the serial port also enables the A/Ds to collect data and triggers the high-voltage pulser. The beamformer uses this serial port pulse for the second and third transmit events. Once the beamformer has finished the entire image, it waits again for a pulse from the motion controller.

Figure 3.8 shows the complete beamforming system. The total system has 64 channels, consisting of 32 VME-based Quad C6201 boards (two channels per board). An additional board, at the end of the chain, provides DC offset corrections, sums data over multiple transmits, performs envelope detection calculations, log compression, and interfaces with a PC for scan conversion. The system is scalable, though additional precision or more complex algorithms would be required to significantly increase the channel count. The system uses two 21-slot VME card cages to house the majority of the beamforming hardware. The prototype system uses two personal computers. One PC provides the ability to load programs, load coefficient data, and debug the beamforming software. This PC contains a PCI based board that attaches to a similar interface in the VME crate to form a PCI-to-VME adapter (Model 617 Adapter, SBS Technologies Inc., St. Paul, MN). The second PC is the control and display PC. The display PC is attached to the beamformer through a FPDP interface

that is mounted on a PCI based DSP board that also has a TI C6201 processor (Model DM 11, Transtech DSP, Ithaca, NY). The result is a greater than 100 MB/s connection between the beamformer and a large circular memory buffer on the display PC. A hardware texture mapping enabled graphics card in the display PC is used for scan conversion and subsequent display.

3.4 Software Implementation

The beamforming software was written in hand optimized assembly language to ensure high performance beamforming. When programming the TI C6201 there are several hardware and assembler constraints that must be considered. The internal data memory of the C6201 is 64 kB of SRAM. The access time for a single word is one CPU clock cycle. Both sides of the CPU and the DMA can simultaneously access the internal data memory if the block/bank architecture is exploited. Algorithms developed for this system have been optimized to avoid memory space conflicts.

3.4.1 Interpolating Processor Algorithm Design

Figure 3.9 shows the main tasks for each of the processors on one of the multi-processor boards. This represents the basic repeatable unit of the beamformer, and two beamformer channels are contained on each of the multiprocessor boards. The algorithms are synchronized to allow real-time beamforming. The interpolating processors are labeled C and D on both Figure 3.4 and Figure 3.9. The interpolating processors are both connected to their particular A/D by an external memory FIFO. The basic task of these processors is to read the data from the A/D and properly

calculate the delayed and apodized values, which are then sent to the appropriate summing processor. For each beamformed point, the delayed value is calculated using the linear interpolation scheme previously described.

There are five basic tasks associated with the linear interpolation processors. These tasks are shown on Figure 3.9 in the relative order in which they are accomplished. Three tasks involve transferring data and are done using one of the DMA channels on the C6201. These I/O tasks are reading the digitized values from the FIFO connected to the A/D packaging logic, reading the coefficients and addresses for the next set of interpolated values, and writing the delayed values to an inter-processor FIFO that is connected to the summing processor associated with the particular interpolating processor. The other two tasks for the interpolators are CPU intensive. They are calculating the interpolated delay values and reordering the digitized values while correcting for DC offsets. As seen in Figure 3.9, the DMA based tasks occur simultaneously with the CPU based tasks.

The algorithm for calculating the delayed values using interpolation has been optimized for the hardware. To calculate a linearly interpolated value, the algorithm must do several things. First, it must load the address of the first digitized value to be used in the interpolation. It must also retrieve the particular coefficients that will be used as multipliers for the interpolation. These coefficients have been pre-calculated for all of the beamformed points, for each transmit, and for each channel. Once the algorithm has determined which digitized values to use in the interpolation, it must load those values into the appropriate registers, multiply them by the previously loaded coefficients, and shift the result to avoid overflow of the 16-bit summing

buffers. For purposes of envelope detection, the beamformer must calculate values for inphase (I) and quadrature (Q) components. This will be discussed further in section 3.7. The I and Q values are packed together as a single word. It is much more efficient to transfer data in large blocks of 4-byte words, and thus packaging the I and Q values improves the efficiency of transferring these values from the interpolating processor to the summing processor.

The entire process of calculating the linearly interpolated pairs of values has been optimized so that each IQ pair requires only 5 cycles of the CPU clock, which is 25 nanoseconds per pair. In order to achieve this rate, the hand optimized assembly code exploited the multiple block and bank architecture for the on chip SRAM.

Figure 3.10 summarizes the organization of the internal SRAM for the interpolating processors. The digitized data is stored in one block of the SRAM, separate from the other major buffers used. There are two buffers that hold the digitized data. In an attempt to simplify the linear interpolation algorithm, these buffers are contiguous in memory. One buffer is used to store the new data that is transferred by the DMA, while the other buffer holds data that is currently being processed by the linear interpolation algorithm. By using two memory buffers, the linear interpolation algorithm can operate independently of the DMA transfer of new data. An additional advantage of using dual buffers is that the data can be collected more quickly than it can be processed. This minimizes the time between transmits and reduces motion artifacts. Unfortunately, it is possible that the linear interpolation algorithm and the DMA transfer of new digitized data could create a memory conflict,

since both may be accessing the same bank in the same block simultaneously. In such a case, the CPU takes precedence and the interpolation algorithm is not disrupted. The DMA is stalled for the single cycle, but there is more than enough time to transfer the new digitized data before the linear interpolation algorithm is complete. The memory buffers used to store addresses and coefficients, and the buffers used to temporarily store calculated values before they are sent to the summing processor, are located in the block opposite of that used for the digitized data. This avoids any memory conflicts that may otherwise occur during the loading of coefficients, the storage of results, or the loading of new digitized values.

Two local buffers are used to store the addresses and 9-bit coefficients needed for linear interpolation. The addresses and coefficients are pre-calculated for each beamformed point, for each transmit, and for each channel. Since the digitized data buffers hold 8000 samples, 13 bits are needed to address all of the samples. Figure 3.11 shows how the address and both interpolation coefficients can be incorporated into a single 32-bit word. For each channel, 120 kB is required to store all of the coefficients for a 10,000 point image. This means that the coefficients must be stored in the large external SB-SRAM and transferred to local buffers, in smaller blocks, for efficient linear interpolation. Using two separate local buffers allows the DMA to transfer the next set of coefficients while the CPU is using the current set.

The hardware for the system has been designed for big-endian operation. Big-endian and little-endian are two standards for data ordering. For big-endian operation the order of the half-word digitized samples in memory is staggered (1,0,3,2,5,4,...). A simple change in the coding for the packaging of the digital samples would correct

this problem. However, for the prototype system, the order cannot be corrected. There are two possible software solutions to this problem. One is to modify the linear interpolation algorithm to account for this staggered storage. However, the overhead generated by this would reduce the performance of the system. Instead, the data can be systematically reordered. The reordering algorithm uses the CPU rather than the DMA. In addition to reordering the data, a pre-determined offset is subtracted from the values. This combined with a calibration allows correction of any DC offset in the digitized data. The reordering and DC correcting algorithm runs simultaneously with the DMA transfer of values from the interpolating processor to the summing processor and does not significantly affect the performance of the system

3.4.2 Summing Processor Algorithm Design

The processors labeled A and B on Figure 3.4 and Figure 3.9 are part of the summing pipeline. There are four basic tasks that each of the summing processors must complete. These tasks are shown on Figure 3.9 in the relative order in which they are accomplished. Three tasks involve transferring data and are done using one of the DMA channels on the C6201. These I/O tasks are reading the data from the interpolating processor, reading the partial sums from the previous channel, and writing the newly summed data to the next channel. Depending on which summing processor is being examined, the data from the previous channel either comes from the FIFO connected to the FPDP or from the inter-processor FIFO connected to the other

summing processor on the board. Likewise, the output FIFO may be connected to an FPDP or to the other summing processor. The remaining task is to sum the data from the previous channel with the data from the current channel.

The internal data SRAM of the summing processors is also organized in a manner that avoids memory conflicts between the two sides of the CPU and the DMA. Figure 3.12 shows the organization of the internal data SRAM for the summing processors. There are three kinds of buffers used by this processor: sum-out buffers, sum-in buffers, and skew buffers. The skew buffers are used to store the interpolated delay values that are received from the associated interpolating processor. The sum-in buffers hold the partially summed values from the previous channel. The sum-out buffers hold the newly summed values. The multiple skew buffers store the partial sums until the appropriate sum has propagated through the pipeline. All of the tasks shown in Figure 3.9 are repeated for each block of data processed. For any given iteration the skew buffer is selected so that the data is summed appropriately. Once the skew buffer is selected, the sum-in and sum-out buffers are located in different banks of the block opposite the skew buffer so that the summing algorithm can simultaneously access any pair of the three buffers without conflict.

The summing algorithm has been optimized so that IQ pairs can be summed in 8 CPU cycles. The code takes advantage of the packaging of the I and Q values in a single 32-bit word and uses a special summing instruction on the C6201 to sum both the I and Q values in a single CPU cycle. The CPU intensive summing algorithm is not done in parallel with any of the DMA transfers of data because the sum is calculated as soon as all the necessary values are available. If the sum were to be

calculated in parallel with any of the DMA transfers, the results would have to be delayed. This would require either more memory or reduced block size and slower I/O.

3.4.3 Synchronizing the Algorithms

Figure 3.9 shows the tasks of each of the processors on the multiprocessor board. The system is synchronized using roughly matched algorithms and limited handshaking. An estimate of the total number of clock cycles used by the interpolating processor is given below. The linear interpolation algorithm requires 5 clock cycles per beamformed point, and an additional 20 cycles of overhead. For 300 beamformed points, the linear interpolation algorithm requires 1520 cycles. It takes 115 cycles of overhead and 2 cycles per word to read the data from the A/D. If 230 words are read each time, the total time is 575 cycles. Reading the coefficients takes 115 cycles of overhead and 1 cycle per word. For a block size of 300 words it takes 415 cycles to read the coefficients. The total number of cycles to read the data from the A/D and the coefficients is 990. Since the linear interpolation algorithm runs in parallel with these transfers, the longer time for the interpolation algorithm determines the total number of cycles for these tasks. After the interpolation is complete, the transfer of the interpolated values to the summing processor requires 115 clock cycles of setup and 600 cycles to complete. The reordering algorithm that executes in parallel with this transfer takes less time and is not the limiting factor. This gives the total time for the interpolating processors as 1520 + 715 = 2235 cycles.

A similar calculation can be done for the summing processors. The summing processors have three DMA transfers, each of which require 2 cycles per word and about 115 cycles of overhead. For these transfers, each with 300 points, the total time is about 2145 cycles. The summing algorithm itself takes about 490 cycles to calculate 300 sums. The total for the summing processor is then 2145 + 490 = 2635. This is roughly matched to the interpolating processor, though the I/O of the summing processor controls the speed of the system. If an image contains 10200, points then there are 34 blocks of 300 points each. Since there are three transmits per image the total number of cycles for a complete image is 2635 x 34 x 3 = 268770. For the five nanosecond cycle time, each image takes roughly 1.3 msec and about 750 images per second can be beamformed. This rate is reduced by the delays required to synchronize the A/Ds on the 64 channels, and synchronizing the data collection with the rocking motion of the transducer will further reduce the frame rate.

The synchronization of the system is done using flags and status registers. At the beginning of each transmit event, the interpolating processors wait for data from the A/D to enter the FIFO attached to the interpolating processors. A flag is set by the FIFO's logic when a threshold number of words in the FIFO is exceeded. This flag is poled by the interpolator until it is set. Once the flag is set, the interpolating processors are free to begin the calculation of the interpolated values. Using this method, the interpolating processors wait for an appropriate start pulse from the transducer on the first transmit event of an image and then wait for a pulse generated by the beamformer each additional transmit event.

The synchronization of the summing processors with the interpolating processors is also done using FIFO flag logic. The interpolating processor will not write values to the inter-processor FIFO connected to the summing processor unless the FIFO-empty flag is set. This flag is set after the summing processor has read the block of data that was to be transferred. This ensures that the interpolating processor does not get ahead of the summing processor. The summing processor will not read values from the inter-processor FIFO connected to the interpolator until the FIFO-almost-full flag is set. This flag is set once a threshold number of data points are present and ready to be read by the processor.

A similar set of FIFO-empty and FIFO-almost-full flags are used to keep the two summing processors on the same board in sync with each other. A slightly different mechanism is used to synchronize the summing processors over the multiple boards. The multiple boards are connected to each other through a FPDP ribbon cable interface. Processor A serves as the FPDP transmitter and Processor B of the next board serves as the FPDP receiver. There are two general purpose I/O bits on the FPDP interface that can be configured to allow communication to occur in the direction opposite to the flow of data. Processor B is able to set or clear each of these bits, and processor A of the previous board is able to read the values of these bits. One of these general purpose I/O bits is used to synchronize communications between the boards. Processor A will not write any data to the FPDP port until the I/O bit is toggled by processor B of the next board. The receiving processor will not read data until a threshold amount is present in the FIFO connected to the FPDP. Determining

if the FIFO holds a number of points greater than the threshold is done using one the FIFO flags. The I/O bit and the FIFO flag together allow multiple boards to be synchronized.

3.4.4 Minimizing I/O

The beamforming software is designed to minimize I/O, which ultimately limits the speed of the system. The I/O for the linear interpolation algorithm can be minimized by restricting the A/D sampling frequency to an integer multiple of the center frequency of the transducer array. This provides samples that are separated by exactly one quarter period and ensures that the linear interpolation coefficients are the same for I and Q. In addition only a single address needs to be stored, as the samples required to calculate I and Q will always be separated by a fixed number of samples. Using this sampling scheme, the delay address and the linear interpolation coefficients for both I and Q can be packed into a single 32-bit word. In addition, packing the linearly interpolated I and Q values together in a single 32-bit word reduces the I/O between the interpolating and summing processors. Restricting the summing buffers to 16 bits also helps to reduce I/O by allowing the I and Q sums to be packaged together in a single 32-bit word and by allowing the use of a special add instruction on the C6201, which sums both the I and Q values in a single cycle.

3.5 Summing Data Over Multiple Transmit Events

The last multi-processor board in the chain shown in Figure 3.1 is responsible for summing beamformed data from different transmit events. In addition to

summing, the board must shift the values to avoid overflow. The algorithm for summing over transmits has been optimized to allow the shifting, summing and repackaging to occur at an average of 2.5 CPU cycles per beamformed point. The summing is done by processor B of the final board and the algorithm is shown in Figure 3.13. The four tasks for this processor are: 1) read the beamformed sum for the current transmit and current block of points from the FPDP interface, 2) read the stored sum over transmits from the SB-SRAM buffer that will be used for the next sum, 3) sum the previously stored partial sum with the current values read from the FPDP, and 4) write the newly calculated sum to an SB-SRAM buffer or to the inter-processor FIFO if all the transmits are complete. In order to reduce limitations on image size, the summing buffers are stored in external SB-SRAM. The additional I/O overhead is not a limiting factor in this situation.

Figure 3.14 shows the organization of the internal data SRAM for this final transmit summing processor. There are three sets of local buffers that are used by the algorithm. The sum-out buffers temporarily store the results of the summing algorithm before they are transferred to SB-SRAM or to the inter-processor FIFO. The in-from-FPDP buffers store the beamformed sums, for a transmit, which were read from the FPDP connected to the summing pipeline of the beamformer. The previous-sum buffers are used as temporary local storage for the partial sums that are read from SB-SRAM. Having multiple copies of the buffers allows the algorithm to avoid memory conflicts. The sum-out buffers are located in an SRAM block separate from other two sets of buffers. This is required because the summing algorithm has the two sides of the CPU accessing the sum-out buffer simultaneously with the other

buffers at different points in execution of the algorithm. Separating these buffers avoids a memory stall by the CPU and allows efficient calculation of the data values. Having two buffers allows the CPU, in the course of its sum calculations, to access one of the buffers while the DMA simultaneously transfers the stored partial sums from SB-SRAM to the previous-sum buffer in the opposite bank.

Synchronization of the final summing algorithm with other processors in the system is done using a method similar to that used to synchronize the multiple boards involved in the pipelined summing. A flag is set by the FIFO logical for the FIFO connected to the FPDP interface. This FPDP interface is connected to the previous board just as any other board in the chain. Once the FIFO has received a threshold number of words from the FPDP transfer, the final summing processor is allowed to read those values from the FIFO. Again, a general purpose I/O bit is used to synchronize the transfer with the previous board. The final summed I and Q values for all transmits are sent to the D processor on the final board. The D processor removes any DC component in the I and Q sums. D also squares the I and Q values and sums the squared values. The sums of the squares are sent to processor A on the same board. Processor A is responsible for the square root and log compression of the signal. The transfers from processor B to D and from D to A are also governed by FIFO flag logic. In this way, the final summing processor is kept in synchronization with the entire beamformer.

3.6 Avoiding Overflow

When designing the algorithms for delaying and summing, care must be taken to avoid overflow at all stages of the processing. The linearly interpolated values for I and Q are packaged together in a single 32-bit word. This limits the I and Q values to 16 bits. In addition the I/O limitations also limit the sums for I and Q to 16 bits each. Because of these limitations the I and Q values calculated by the linear interpolation algorithm must be truncated. The number of bits required to calculate the linearly interpolated values without truncation can be determined. The coefficients for linear interpolation are limited to 9-bit unsigned values by the need to package the coefficients and address information in a single 32-bit word. Thus the maximum coefficient value is 512. The A/D digitizes with 12-bit resolution and the hardware zero extends to 16 bits. The maximum positive value for the 16-bit, zero-extended, digitized value is 32752. The maximum product of coefficient and the digitized value is 512*32752 = 16769024, which requires 25 bits to store. The linearly interpolated value is the sum of two of these 25-bit products. 26 bits would be required to hold the linearly interpolated value if the coefficients were arbitrary 9-bit unsigned numbers. But, since the coefficients are weights to interpolate between two data points, the sum of the coefficients is less than or equal to 512. This means that the sum of the products of the coefficients and digitized values can be stored in a 25-bit number. Apodization reduces both coefficients and, therefore, the 25-bit number is also adequate in the case of the apodized coefficients.

Since the interpolated values will be summed over the 64 channels of the system, the 25-bit value must be truncated to avoid overflow in the 16-bit summing

buffer. At first glance it appears that the 25-bit values must be truncated to 10 bits to avoid overflow. Since 2^6=64, summing 10-bit values over the 64 channels results in a 16-bit sum. However, if apodization is considered the truncation can be made less severe. A raised cosine receive apodization reduces the maximum sum by a factor of one half. As a result, the 25-bit interpolated values can be truncated to 11 bits, instead of 10 bits, without overflow.

It should be noted that for the purposes of the synthetic aperture beamformer described in this work, there is no apodization applied to the transmit pulses. All apodization is applied in software by adjusting the linear interpolation coefficients. However, it is still natural to think of the weighting of the receive signals across the channels as the receive apodization, and the additional weighting, applied on receive, for the different transmit events as transmit apodization.

For each transmit event, the sum over the 64 channels of the I and Q values will be restricted to a 16-bit number by the shifting described above. The I and Q sums for each transmit event must be added by the final processing board. This sum is also restricted to 16 bits. This restriction is due in part to the I/O limitations of the system, but there is also the practical matter of calculating the estimate of the signal envelope as the square root of the sum of the squares of the I and Q sums. To square I, the processor will only be able to multiple two 16-bit values, so there is no need to calculate more than 16 bits. For the present system there are three transmit events. In the absence of transmit apodization, each transmit event would 16 bits and the sum of three 16-bit numbers would require 18 bits. However, if the transmit events are sufficiently weighted the sum is reduced to a 17-bit number and the individual

transmit event sums only need to be truncated by 1 bit to be summed in a 16-bit buffer. In this way overflow is avoided in the system, while maintaining the maximum dynamic range.

3.7 Efficient Envelope Detection and Log Compression

The final DSP board at the end of the pipeline is also responsible for envelope detection and log compression. For a fixed-point processor, direct calculation of these complicated functions would be too slow for real-time beamforming. Instead, the real-time beamformer uses a lookup table (LUT) to determine the appropriate 8-bit pixel value.

The envelope of the signal can be estimated using second order sampling. If we assume that the beamformed signal has the form shown in equation (2), and that the envelope is slowly varying then second order sampling can be used to estimate the envelope.

$$s(t) = A(t)\cos(\omega_o t + \phi) \qquad (2)$$

where A(t) is the envelope, s(t) is the signal, ω_0 is the angular frequency, and ϕ is the phase. A second signal, calculated at a time $t + \delta$ (where $\delta = T/4$ and T is the period corresponding to the angular frequency ω_0), is shown in equation (3).

$$s(t+\delta) = A(t+\delta)\cos(\omega_o t + \phi + \omega_o \delta) \qquad (3)$$

If the envelope is slowly varying, $A(t) \cong A(t+\delta)$. Since $\delta = T/4$, $\omega_0\delta = \pi/2$ and equation (3) can be written as follows:

$$s(t+\delta) \cong A(t)\sin(\omega_0 t+\phi) \qquad (4)$$

Letting I = s(t) and Q = s(t+ δ) we can estimate the envelope signal as:

$$A \approx \sqrt{I^2+Q^2} \qquad (5)$$

The algorithm employed on the fixed-point processor must calculate this envelope estimate, as well as log compress the values. The inputs to this algorithm are the I and Q sums, each of which are 16-bit values. The fixed-point processor can easily calculate the sum of the squares of the I and Q values by simple multiplication and addition. The result is a 32-bit value. A lookup table can be created, which gives the value of the log of the square root of the input. However, a lookup table with the results for all of the possible 32-bit values would require four gigabytes of memory. Such a large table is not possible, and also not necessary to produce an output value which has only 8 significant bits. In order to determine the necessary number of bits, the radiation pattern for a point target was simulated and lookup tables of various sizes were compared with a floating-point calculation. Simulations of the radiation patterns for varying numbers of bits showed that a 21-bit lookup table ensures that the radiation pattern for the fixed-point system is less than 1 dB different than radiation pattern calculated using floating-point mathematics. However, a 21-bit lookup table would still require over 2 MB of memory, which is more than is available in the SRAM of the TI C6201. A 14-bit lookup table requires only 16 kB, but severely limits the dynamic range of the system. Figure 3.15 shows the radiation pattern

generated for a point target at F/4 for the system using floating point math and using a 14-bit lookup table. The secondary lobes for the radiation pattern using a 14-bit lookup table are less than 40 dB below the peak. A compromise, which meets both the accuracy and memory requirements, is a dual resolution lookup table. Figure 3.15 shows that the dual resolution table has a radiation pattern which is nearly that of the 21-bit lookup table. The dual table, implemented in this system, uses a 14-bit lookup table for all but the lower 1000 values. For these lower 1000 values the appropriate portion of the 21-bit lookup table is used. The memory required for the dual lookup table is less than 17 kB and the accuracy is less than 1 dB different than the floating point calculations. The secondary lobes are 50 dB below the peak for the dual lookup table. Using hand optimized assembly code, this dual resolution LUT provides an 8-bit estimate of the log compressed envelope in less than 25 nanoseconds per point.

3.8 Conclusions

The design for a beamformer capable of producing over 6.6 million points per second was presented. The beamformer is limited by the I/O capacity of current state of the art DSP hardware. However, by restricting the sampling rate of the A/D to reduce coefficient storage, packing data for efficient transfer, and carefully managing memory resources, this bottleneck can be reduced. The system is capable of calculating IQ pairs in 25 ns and summing IQ pairs in 8 ns. A dual resolution lookup table provides real-time envelope detection and log compression.

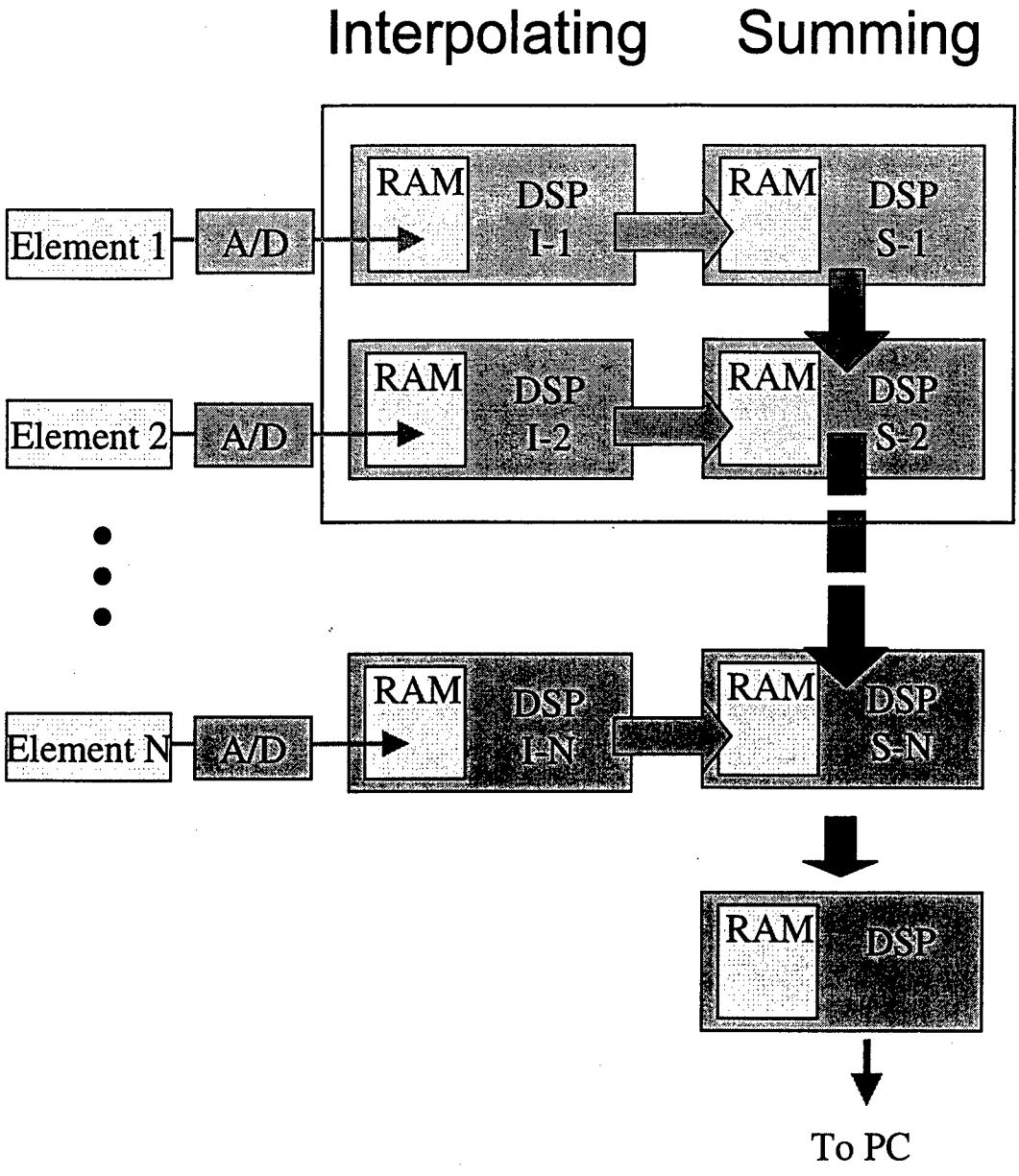


Figure 3.1 Digital signal processing network with an interpolating processor and a summing processor for each channel of the system. The summing processors are arranged in a pipeline structure.

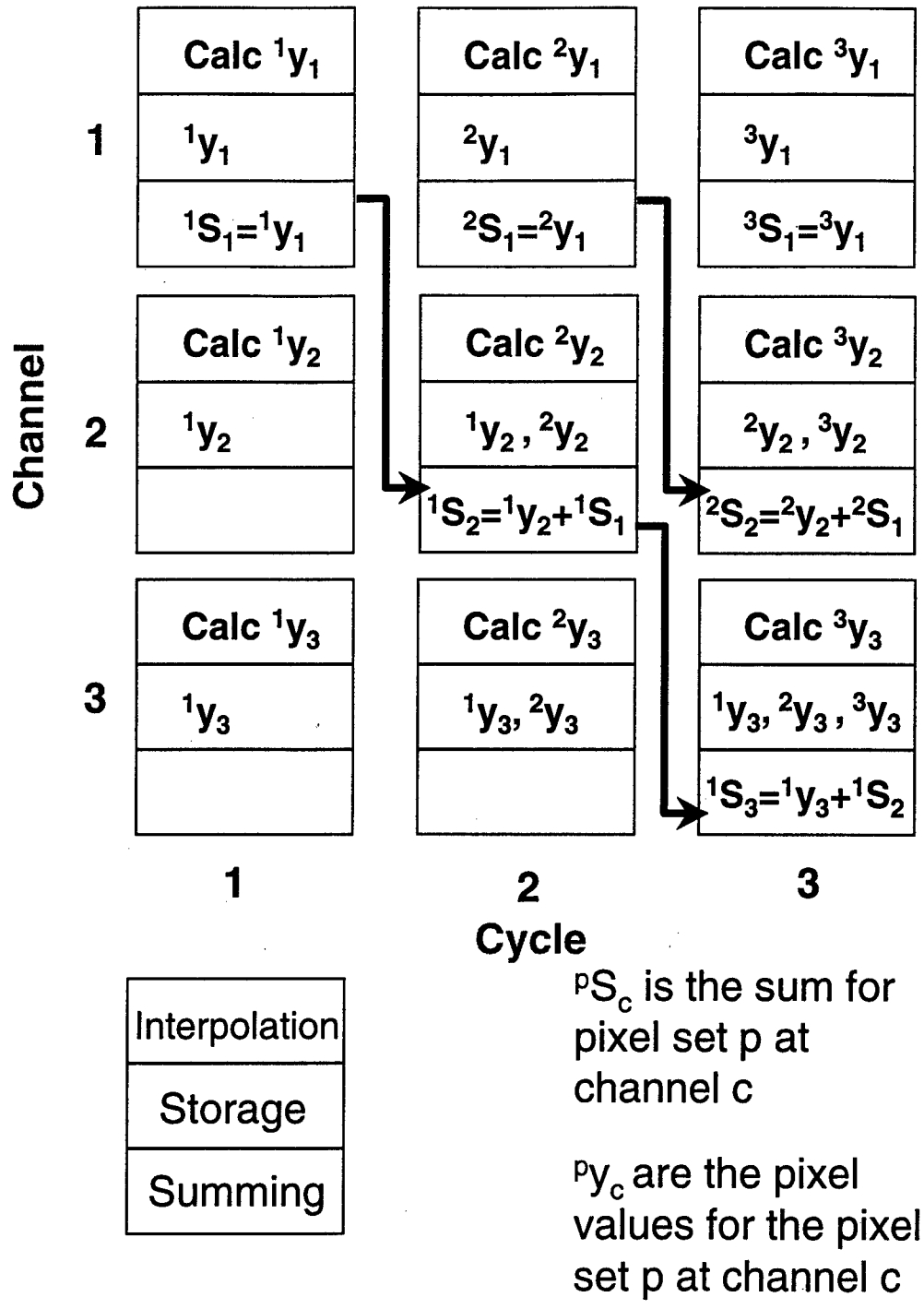


Figure 3.2 Illustration of data skew and required data storage.

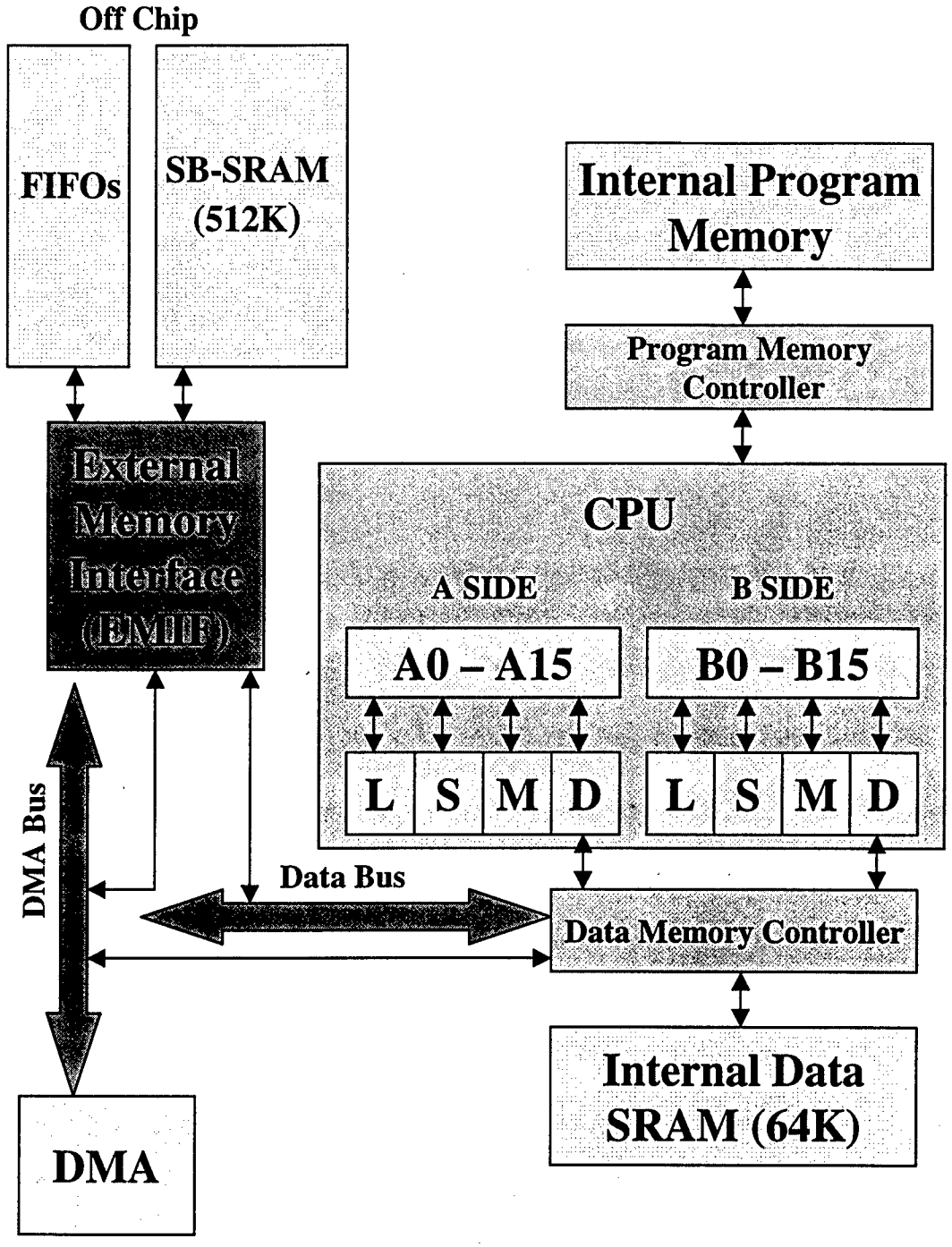


Figure 3.3 Architecture for the Texas Instruments TMS320C6201 digital signal processor.

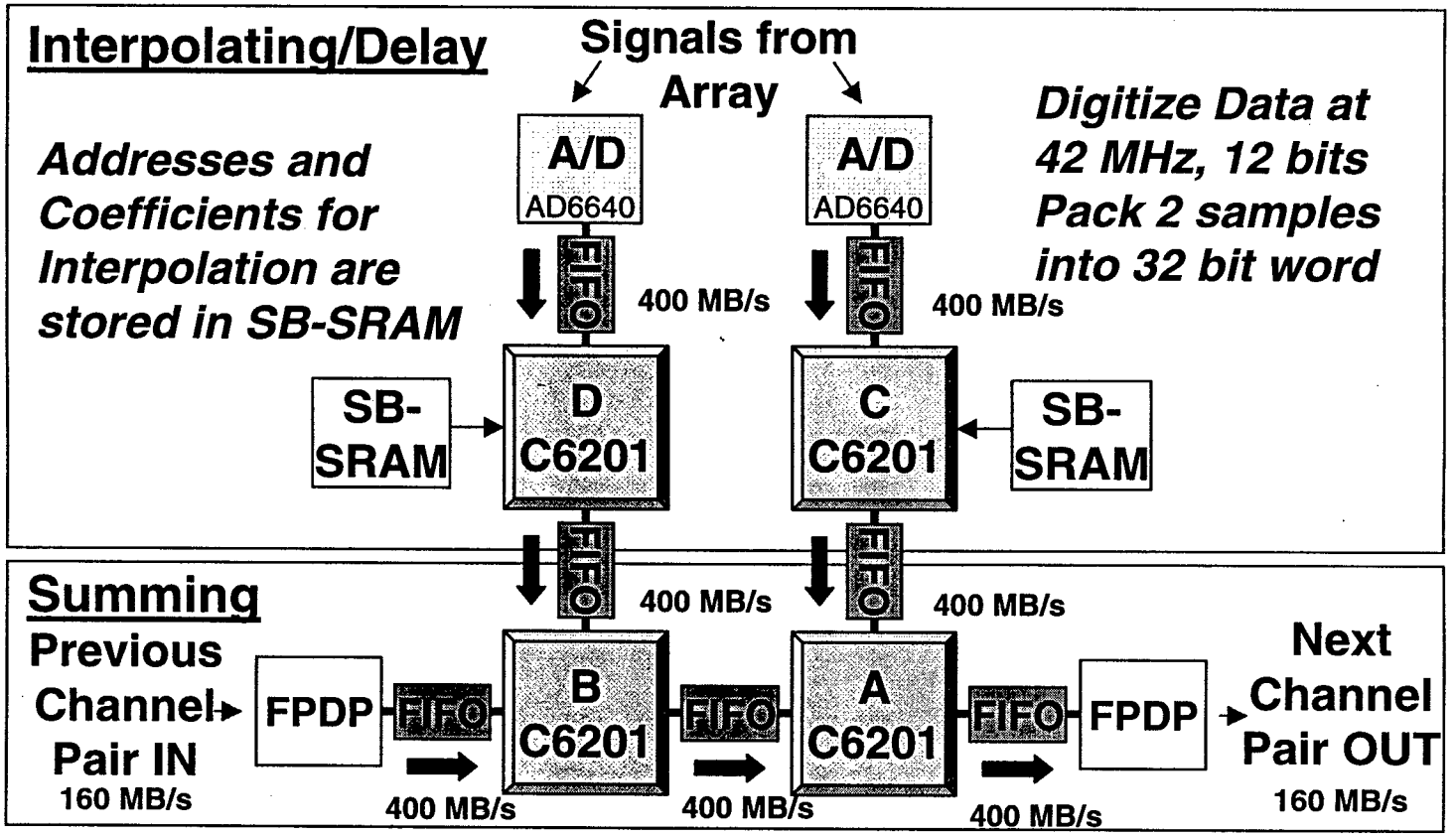


Figure 3.4 Multiple processor board from Pentek. Two channels of the system are contained on each board. These channels consist of an A/D, an interpolating processor, and a summing processor.

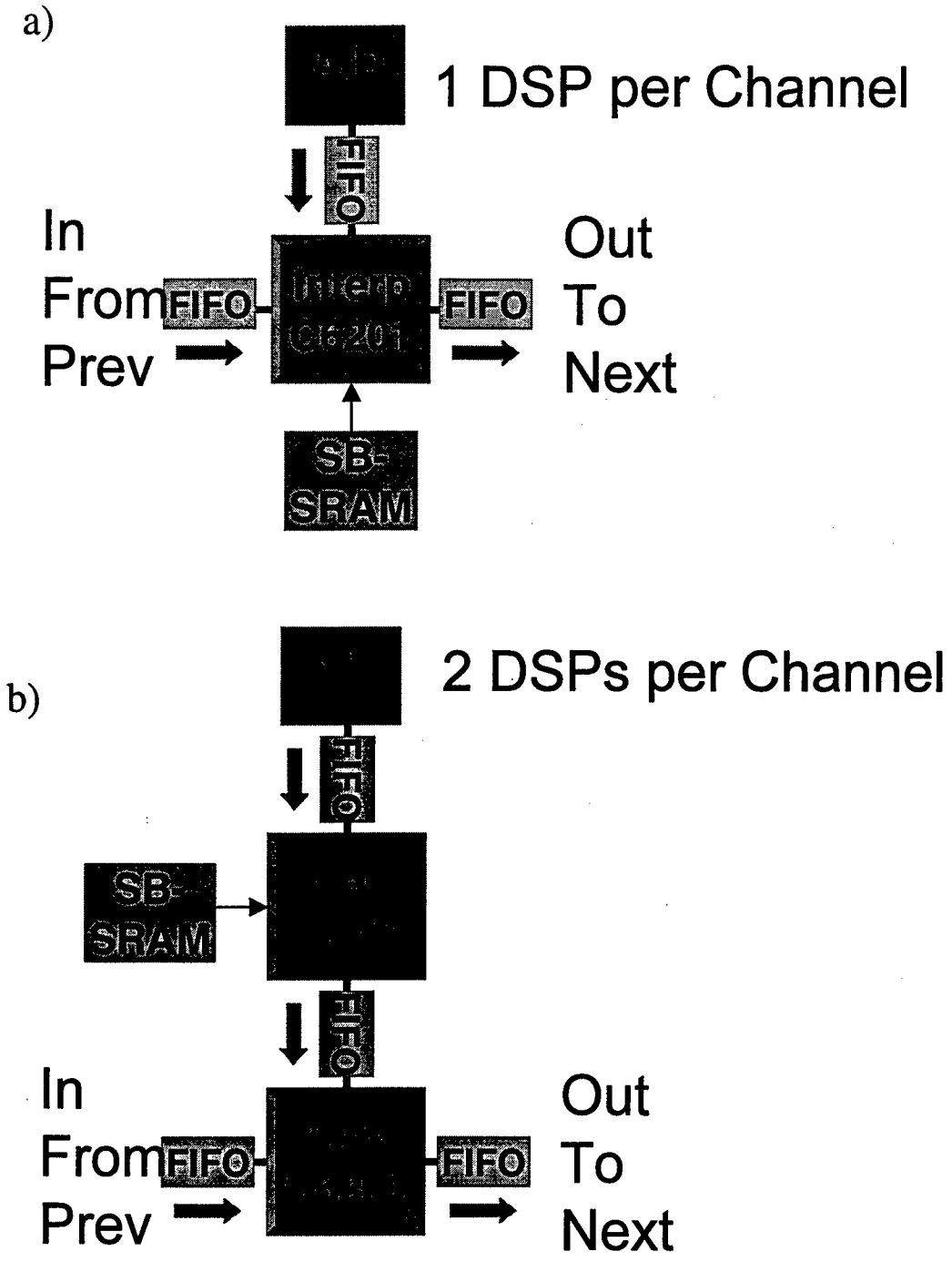


Figure 3.5 a) One processor per channel. b) two processors per channel.

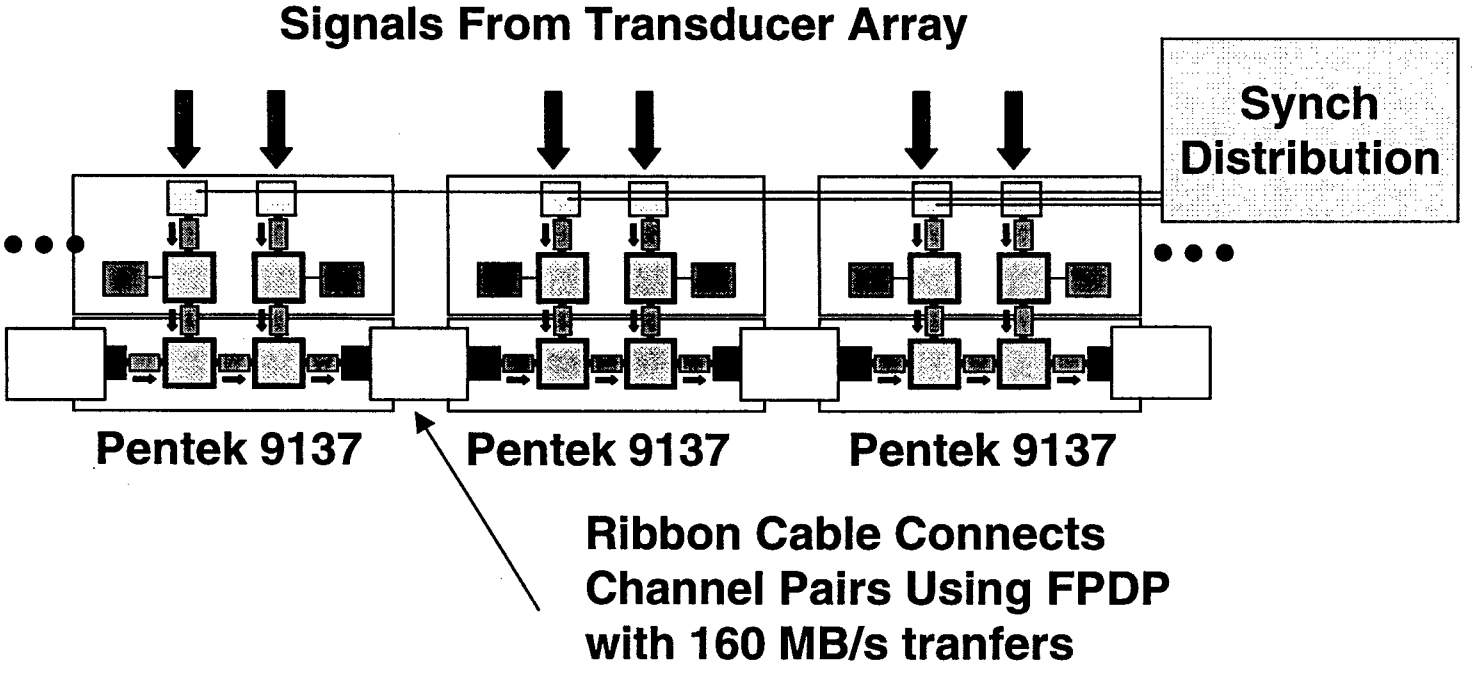


Figure 3.6 Multiple board connections.

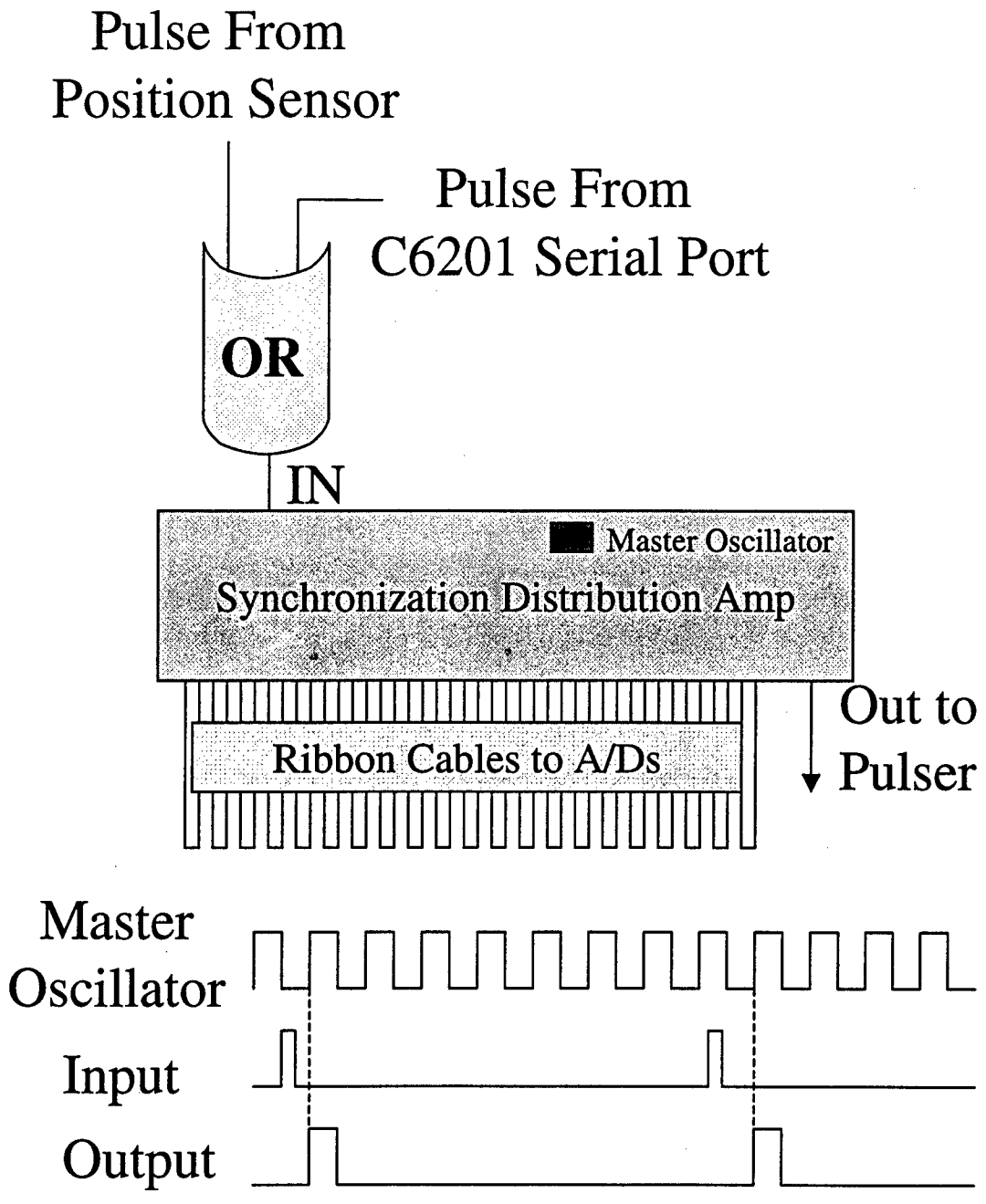


Figure 3.7 Synchronization distribution amplifier.

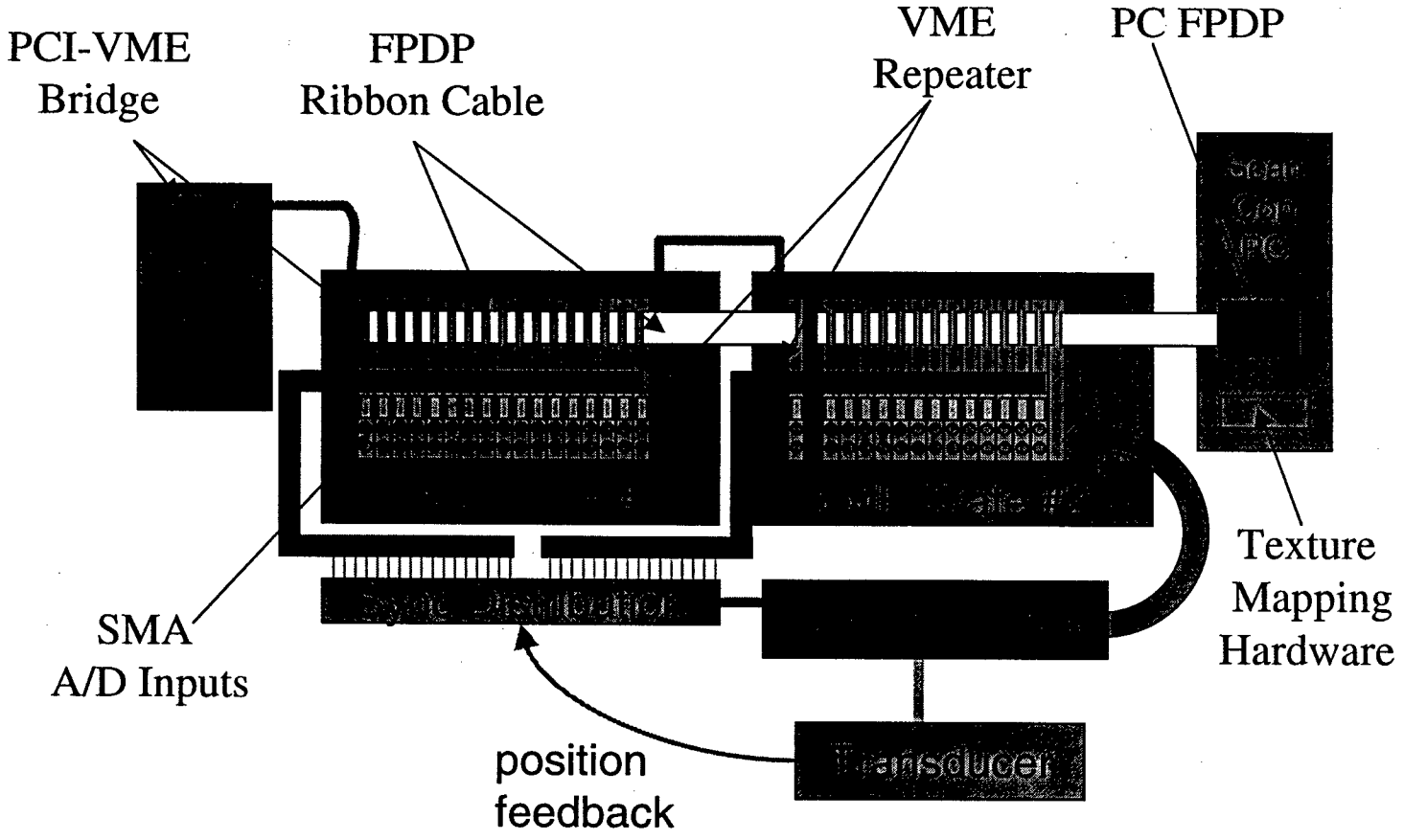


Figure 3.8 Complete beamforming system diagram.

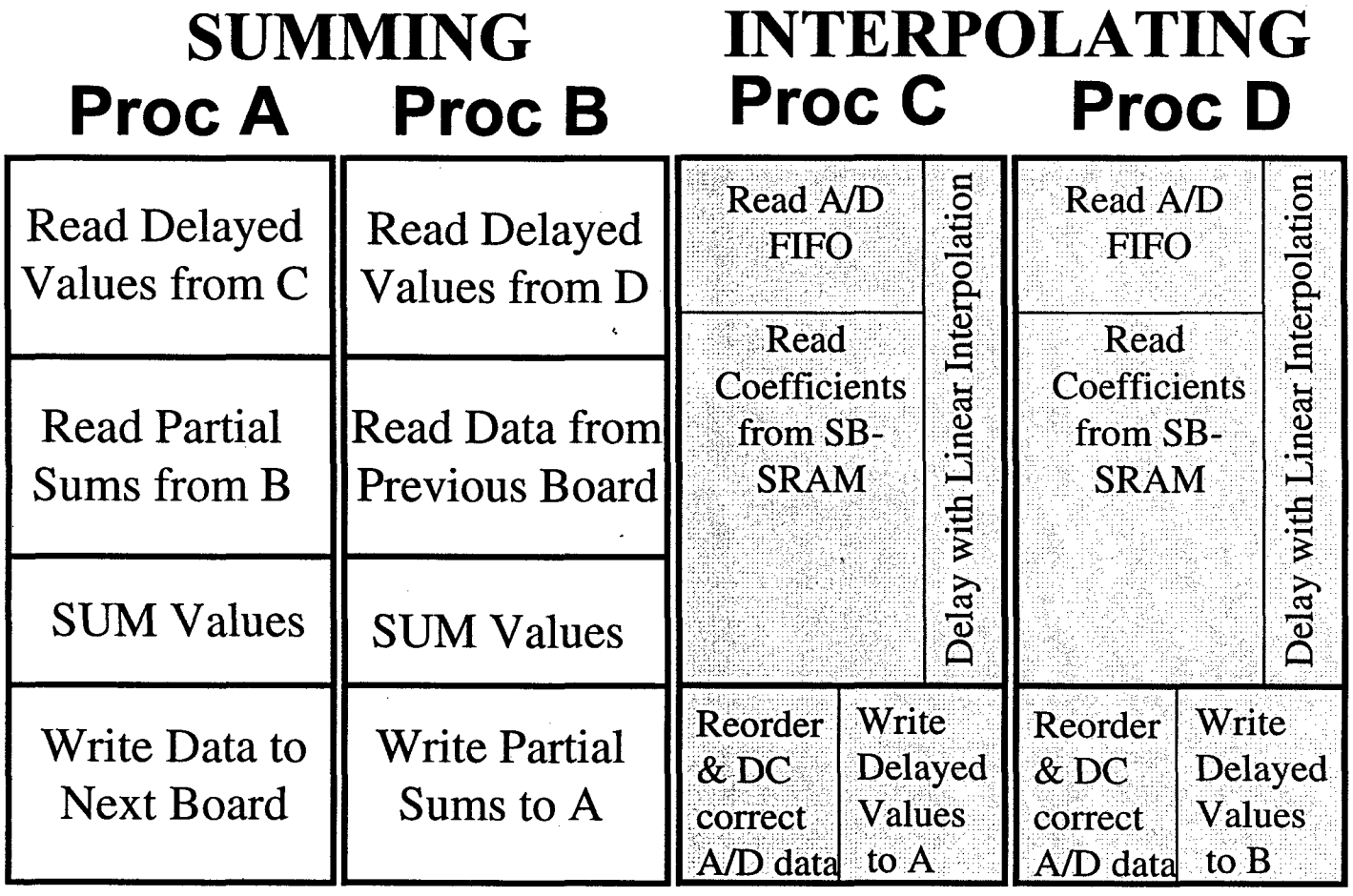


Figure 3.9 Software design for one of the multi-processor boards. This represents the basic repeating unit for the beamformer and contains two beamformer channels.

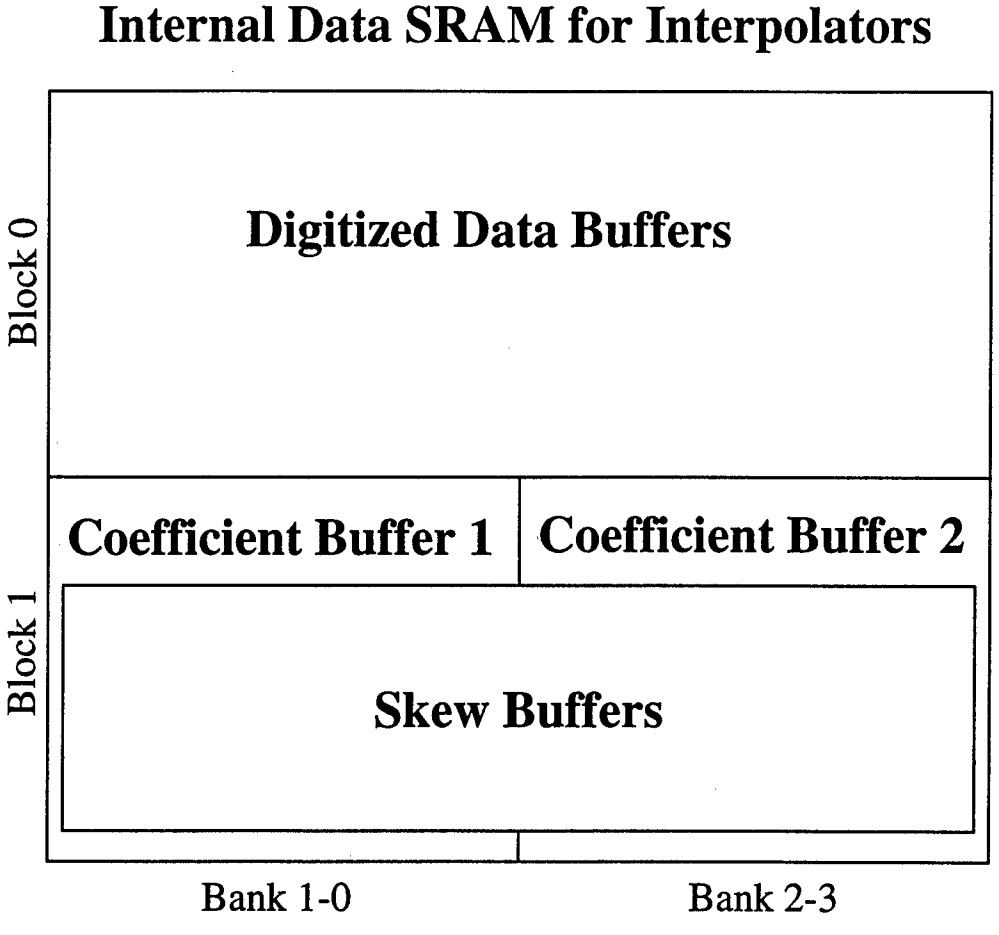


Figure 3.10 Internal SRAM organization for interpolating processors.

Figure 3.11 Packaging of address and two 8-bit coefficients in a single 32-bit word.

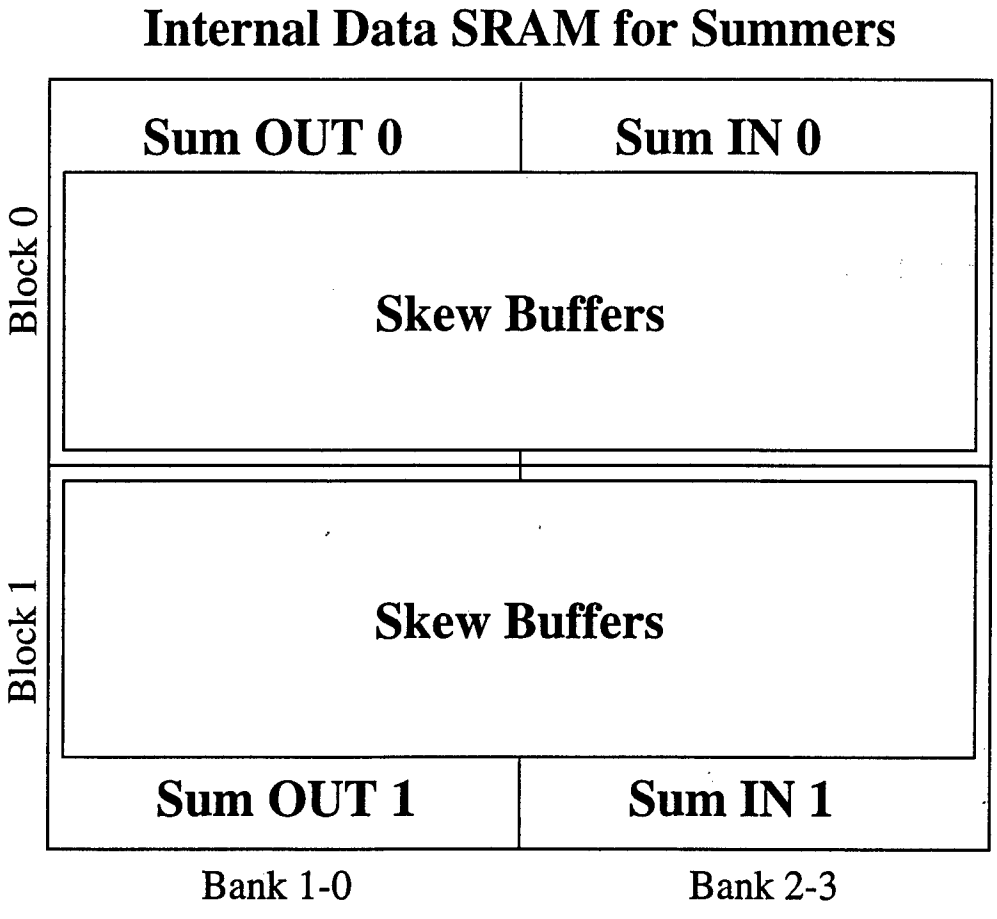


Figure 3.12 Internal SRAM organization for summing processors.

Algorithm for Final Summing Processor B

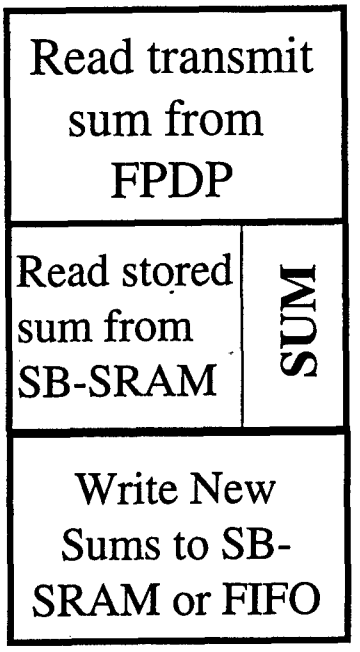


Figure 3.13 Algorithm for summing over multiple transmits on processor B of the final multi-processor board.

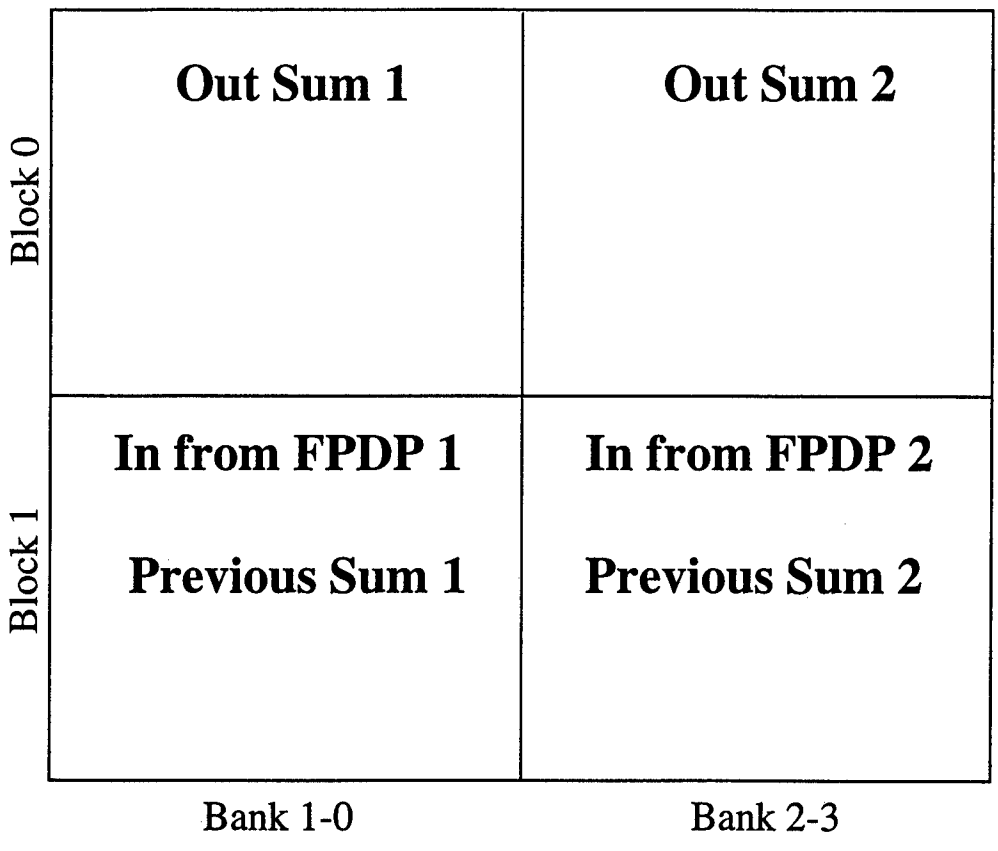


Figure 3.14 Internal SRAM organization for the processor B on the final board.

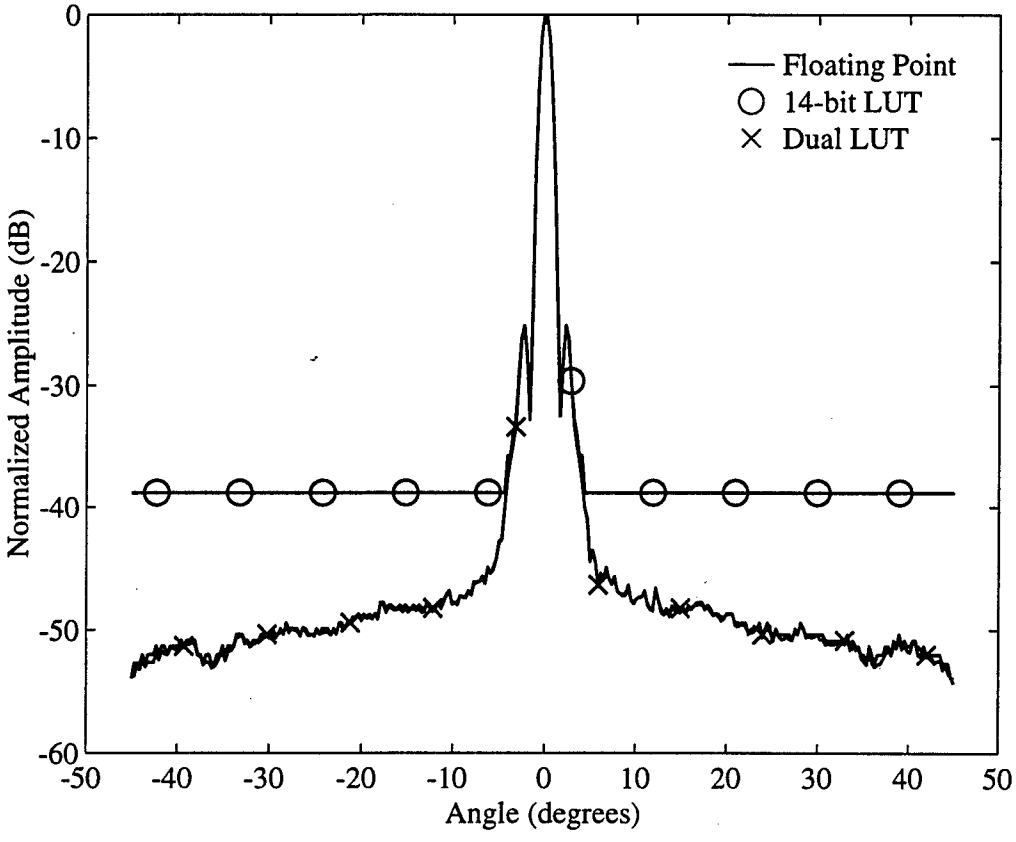


Figure 3.15 Radiation pattern for system using floating point math, a 14-bit LUT and a dual resolution LUT. All results are simulated.

ENDNOTES

[1] "TMS320C6000 Technical Brief." TI Literature # SPRU197D, February 1999.

Cine MPR: Interactive multi-planar reformatting of four-dimensional cardiac data using hardware-accelerated texture mapping[1]

Raj Shekhar[a] and Vladimir Zagrodsky

Department of Biomedical Engineering, Lerner Research Institute
The Cleveland Clinic Foundation, Cleveland, Ohio 44195, USA

[a]Corresponding Author: Raj Shekhar, Ph.D.
Department of Biomedical Engineering (ND20)
Lerner Research Institute
The Cleveland Clinic Foundation
9500 Euclid Avenue
Cleveland, Ohio 44195

Phone: 216/445-3246
Fax: 216/444-9198
Email: shekhar@bme.ri.ccf.org

[1]This project was supported by the Department of Defense research grant DAMD17-99-1-9034 (PI: Raj Shekhar) and the Whitaker Foundation research grant RG-01-0071 (PI: Raj Shekhar).

ABSTRACT

Four-dimensional (4-D) imaging to capture the three-dimensional (3-D) structure and motion of the heart is an emerging trend. We have presented here interactive multi-planar reformatting (MPR), i.e., the ability to visualize any arbitrary anatomical cross-section of 4-D cardiac images and to change its orientation smoothly while maintaining the original heart motion. Continuous animation to show the time-varying 3-D geometry of the heart and smooth dynamic manipulation of the reformatted planes, as well as large image size (100-300 MB), make MPR challenging. Our solution exploited the hardware acceleration of 3-D texture mapping capability of personal computer graphics boards. Novel uses of volume subdivision and caching concepts allowed us to use this hardware and further improved the efficiency of our solution. We were able to visualize and smoothly interact with real-time 3-D ultrasound cardiac images at the desired frame rate (25 Hz). The developed methods are applicable to MPR of most 3-D and 4-D medical images, including 4-D cardiac images collected in a gated fashion. Our algorithms also extend nicely to simultaneous display of two 4-D images and provide a generalized framework for more complex visualization tasks.

Key Words: multi-planar reformatting, visualization, four-dimensional cardiac image, real-time three-dimensional ultrasound, texture mapping

I. INTRODUCTION

Four-dimensional (4-D) imaging is the most natural mode for recording both structure and function of the bodily organs. Such spatiotemporal images comprise a sequence of three-dimensional (3-D) snapshots of a moving body part. Since the anatomy may have an inherent motion, as in the case of the heart, or the subject may move during imaging, a complete 3-D image should ideally be collected within a very short time to avoid motion artifacts. Most imaging modalities, however, are too slow to permit real-time 3-D acquisition of 4-D images, and, therefore, rely on patient immobilization and gating.

Real-time 3-D ultrasound, an exciting new development [1-3], is an exception to this generalization. Real-time 3-D ultrasound is powerful in that it can capture the complex 3-D geometry as well as the motion of an organ extremely rapidly. This property makes real-time 3-D ultrasound particularly suitable for imaging the heart. In fact, the real-time 3-D ultrasound, for the first time, has allowed clinicians to image the heart in its entirety within one cardiac cycle (less than 1 second) without requiring cardiac and respiratory gating. Real-time 3-D ultrasound is advantageous even when imaging static anatomy. Extremely short scan times obviate the need to immobilize a patient during imaging and ensure distortion-free 3-D images. When time is of the utmost importance, such as in emergency or battlefield medicine, real-time 3-D ultrasound offers the only quick and accurate means of performing volumetric imaging.

The benefits of real-time 3-D ultrasound and 4-D imaging in general accompany new challenges for visualizing the data they produce. Nowhere else are these challenges greater than when working with cardiac images. When displaying dynamic cardiac data, it is crucial that the original frame rate is maintained so as not to alter the underlying heart rate. Important diagnostic cues are derived from the heart motion; maintaining the original heart motion is, therefore, critical to making an accurate diagnosis.

Conventional ultrasound, although performed in real time, yields two-dimensional (2-D) images. Cardiac cine loops, sequences of 2-D images spanning a full cardiac cycle and showing a specific anatomic cross-section, have historically been saved on videotapes for storage and review. Real-time 3-D ultrasound images cannot be stored on videotapes, which are fundamentally 2-D. Although many hospitals have converted to digital systems recently, various proprietary and nonproprietary movie formats used to store and review images can accommodate only 2-D images. To continue to use existing capabilities in conjunction with real-time 3-D ultrasound, one could extract and save a limited number of representative 2-D views (cine loops) from the 3-D data, but doing so undermines the very strength of real-time 3-D ultrasound. Ideally, real-time 3-D ultrasound images need to be visualized by interactively exposing any of the infinitely many anatomic cross-sections through the time-varying 3-D data.

Displaying any arbitrary cross-section of a volumetric image is commonly known as multi-planar reformatting (MPR) [4-6]. For real-time 3-D cardiac images, the reformatted plane should be animated to show the beating of the heart. We have coined the term "cine MPR" to refer to MPR of dynamic 3-D data. Although shaded surface rendering and volume rendering have been proposed for displaying 3-D images [7], MPR remains the most trusted visualization mode clinically. Shaded surface rendering requires segmentation of the structures of interest, which remains a difficult problem. Because ultrasound images are noisy and lack a direct correlation between voxel intensity and tissue type, opacity curves are hard to define based solely on intensity values, making good quality volume rendering difficult without extensive pre-processing. Simultaneous display of up to three intersecting reformatted planes through the volumetric data remains the most practical visualization mode. Clinicians can explore the time-

varying 3-D geometry and important and "interesting" landmarks by smoothly varying the orientation of the reformatted planes. Interactivity without any perceptible delay is key to the success of cine MPR.

The greatest challenge in interactive cine MPR is the manipulation of large amounts of data in a very short time. Performing interactive cine MPR using the CPU of most modern desktop computers is still not possible because the task is memory access-limited. Therefore, we have developed methods that exploit the hardware-accelerated 3-D texture mapping capability of commercially available high-end graphics boards. This capability has been used for accelerating volume rendering [8-13]. To meet the necessary performance requirement here, we use volume subdivision and caching concepts in novel ways. Although visualizing real-time 3-D ultrasound cardiac images is the focus of this paper, the developed methods are applicable to most 3-D and 4-D medical images, including 4-D cardiac images collected in a gated fashion. Our algorithms also extend nicely to simultaneous display of multiple data sets.

II. COMPUTATIONAL CHALLENGES OF INTERACTIVE CINE MPR AND INTRODUCTION TO 3-D TEXTURE MAPPING HARDWARE

Whether a 4-D image is acquired in real time or in a gated fashion, the challenges for its visualization remain the same. Cine MPR of a single 4-D cardiac image must handle a large quantity of data (100-300 MB) and allow interactive viewing of any anatomic cross-section while maintaining a rate of 10-30 frames per second (fps). Indeed, applications that require visualizing two 4-D images simultaneously double the computational and data handling requirements.

An arbitrary oblique cutting plane through a discrete 3-D image rarely passes through the original voxels defined on a rectilinear grid. Displaying an oblique plane, therefore, involves generating new samples on the cutting plane, necessitating data interpolation. Nearest-neighbor, trilinear and high-order interpolations are various possible interpolation schemes. We have chosen trilinear interpolation, which represents a good trade-off between accuracy and computation and is an accepted interpolation scheme. Overall, cine MPR of a 4-D cardiac image requires up to 25 million trilinear interpolations per second (MTRIPS), which equates to 525 million floating-point operations per second (MFLOPS) and 225 million memory accesses per second (MMAPS). Most modern CPUs are capable of meeting the floating-point operations requirement, but the memory access requirement degrades the performance (achievement of the desired frame rate). Only 12.5 MMAPS are possible, given that the typical random memory access speed is 80 nsec. Since the memory access time does not evolve according to Moore's law, the large disparity between the number of memory accesses required and the number possible (225 vs. 12.5 MMAPS) is unlikely to be bridged in the near future. Moreover, medical images have grown in size with the advances in the computing technology, raising the computational demands further.

To make interactive cine MPR possible, we exploit the capabilities of 3-D texture mapping hardware. The technique of 3-D texture mapping (also known as solid texture mapping) is a standard one used in computer graphics to simulate realistic patterns (texture) like wood or marble [14]. Conceptually, the rendering process treats a 3-D object as if it were carved out of a large block of wood or marble, represented digitally as a 3-D discrete array. Since the polygonal faces describing a 3-D object rarely coincide with the voxels of the 3-D texture, interpolation is needed to generate new samples on the faces of the object. Because 3-D texture mapping is a numerically intensive task, the graphics industry has developed hardware accelerators. MPR

parallels 3-D texture mapping when the volumetric medical image assumes the role of the 3-D texture and the cutting plane, that of the 3-D object. To produce the "cine" effect when visualizing 4-D images, the 3-D frames of the sequence are cycled at the desired frame rate.

The latest 3-D texture mapping hardware can perform over 300 MTRIPS, almost an order of magnitude greater than cine MPR's requirement. To achieve this performance, however, the necessary data must be present in the hardware-accelerated (resident) texture memory. Performance suffers if new data need to be copied to the texture memory "on-the-fly." Because texture memory is generally limited in size, a 4-D image seldom fits in the available resident texture memory completely. The volume subdivision concept we describe below is a means to reduce the data requirement for optimal usage of the limited texture memory. Despite volume subdivision, the reduced data may still exceed the texture memory. In such cases, copying new data to the texture memory during an interactive session cannot be avoided. The custom caching methods minimize such copying to maximize performance.

III. CONCEPTS USED IN THE INTERACTIVE CINE MPR ALGORITHM

A. *Volume Subdivision*

Volume subdivision divides up a 3-D image or a volume into smaller 3-D blocks of equal size. For spatiotemporal 4-D images, each frame is subdivided individually and identically. Block dimensions must be a power of two, a constraint imposed by OpenGL, the graphics library we use. Although the subdivision can be carried out using differing block sizes (8^3, 16^3, 16 x 16 x 8, etc.), the block size must remain the same in a given application for optimal caching.

For volume subdivision and subsequent rendering, we assume that a 3-D image exists in the positive octant of a 3-D coordinate system such that one vertex of the image coincides with the origin. The volume subdivision starts from the origin and is continued along the three principal directions. The use of trilinear interpolation dictates that the blocks must overlap by a single layer of voxels for seamless tiling upon rendering. Therefore, all intermediate blocks share their six faces with the neighboring blocks. The blocks on the boundary of the volume will have fewer than six neighbors and will share only inner faces with their neighboring blocks. To explain volume subdivision numerically, if a 128^3-sized 3-D image were to be subdivided into 16^3-sized blocks, there would be 9 blocks along each direction, factoring in the one-voxel overlap, or a total of 9^3 blocks. In the present example, the last block on the X-axis would have valid data only in the first 8 layers (parallel to the Y-Z plane). Likewise, the block farthest away from the origin would have valid image voxels only in a contiguous 8^3 region. The empty regions of the partially filled blocks are set to zero. Fig. 1 shows volume subdivision schematically.

The foremost advantage of volume subdivision is that it removes the limitation that the original 3-D or 4-D image be smaller than the texture memory. By keeping the block size smaller than the size of the texture memory, data sets of any size can be visualized. We keep the block size significantly smaller than the texture memory size for flexibility and efficiency, which will become apparent later.

A second advantage stems from the fact that the entire volume is rarely required for most visualization tasks. For instance, MPR requires only those voxels either immediately in front of or behind the cutting plane. The performance can be improved if unnecessary data are not copied to the texture memory. Volume subdivision provides the "granularity" to reject unnecessary data in units of blocks, thus lowering the data requirement (Fig. 2). Since trilinear interpolation requires a 2^3-voxel neighborhood, the smaller the block size, the greater is the data reduction.

However, because the blocks must have a one-voxel layer overlap, the duplication of voxels increases with decreasing block size. For instance, in the extreme case of 2^3-sized blocks, all voxels except those on the surface of the original volume are duplicated. If rendering required all blocks, the texture memory requirement would be almost twice the size of the original data. With smaller blocks, the computational overhead also increases, causing performance to suffer. Therefore, neither very large nor very small blocks are ideal for optimum performance. Moreover, no single block size is ideal for all applications. We choose the optimal block size experimentally.

A third advantage of volume subdivision is that the frequently used blocks can be cached in the texture memory more permanently to minimize data transfer. Caching will be explained in section C.

B. Rendering Reformatted Planes

The MPR of a 3-D image is the intersection of a cutting plane, coinciding with the reformatted plane, with the image volume. After the volume has been subdivided, this plane intersects a subset of blocks. Rendering the current view requires only these blocks as long as the orientation of the cutting plane does not change. We will describe our method for a fixed orientation before describing how the interaction is handled.

When a plane intersects a cube, the cross-section is a three-, four-, five- or six-sided polygon. The number of polygons indeed equals the number of blocks that the cutting plane intersects. In our case, the intersection of the cutting plane with the subdivided volume yields a mosaic of polygons. We compute the vertices of each polygon and arrange them in a counter-clockwise order (so that OpenGL can treat them as front-facing polygons and not discard them in the rendering process). Following vertex calculations, we compute 3-D texture coordinates for each vertex of each polygon, treating the associated block as the 3-D texture. The block dimensions are normalized so that the texture coordinates fall between 0 and 1 (Appendix 1 contains further details). Following the calculation of polygon vertices and texture coordinates, the reformatted plane is displayed by rendering the mosaic of polygons while texture mapping them simultaneously. The order in which the polygons are rendered is immaterial because they abut one another, but do not overlap. Fig. 3 shows a reformatted plane in which each polygon is colored differently to illustrate the underlying mosaic and the seamless tiling of the polygons.

For cine MPR, the rendering of the reformatted plane is repeated for each frame of the sequence. As long as the orientation stays fixed, the spatial arrangement of the required blocks within a frame does not change and the calculation of these blocks does not need to be repeated. The previously calculated polygon vertices and texture coordinates are also reused. Successive frames are rendered at regular intervals. When the last frame is reached, we cycle back to the first frame. The original frame rate of real-time 3-D ultrasound decides the desired inter-frame interval to render successive frames. For a typical 25 fps acquisition, the inter-frame interval is 40 msec, and the key to successful cine MPR is to keep the rendering time of a frame below this interval. If the rendering time is less than the inter-frame interval, delays are introduced to maintain the desired frame rate. If the rendering time exceeds the specified interval, the display frame rate will be slower than desired.

If a number of reformatted planes are to be viewed simultaneously, the above process needs to be repeated for each orientation. The blocks needed for each reformatted plane are determined individually. Likewise, the calculation of polygon vertices and texture coordinates are repeated for each view. Some blocks may be common among the views; these blocks are called "common" blocks and are given higher priority in the texture memory. As before, to

maintain the correct frame rate, rendering all polygons for all views must be completed within the inter-frame interval.

When a user interacts and changes the orientation of the reformatted plane(s), the calculation of intersected blocks, polygons and their vertices, and texture coordinates is repeated. Caching controls the storage and updating of the blocks in the texture memory. The other rendering steps remain the same.

C. Caching

A texture must exist in the texture memory before it can be used. All blocks needed for cine MPR, therefore, must be brought into the texture memory first. Since the block size is significantly smaller than the size of the texture memory, thousands of blocks can reside in the texture memory simultaneously. When a specific block is needed, the texture memory is checked first for its existence. If the block does not exist, it is copied to the texture memory, overwriting an existing block. The texture memory thus acts like a cache in our method. How this cache is initialized and updated directly affects performance.

Texture objects serve as place holders for image blocks in the texture memory. Image blocks are copied to the data field of existing texture objects. Because creating and destroying texture objects is time consuming and can adversely affect performance, they are created only once at the beginning. In order for any texture object to accept any block, all texture objects are made identical in size and format, which match those of the blocks. This explains why volume subdivision creates blocks of identical size. As many texture objects as can fit in the texture memory are created. Moreover, each texture object has a unique identifier and an additional field to identify the block currently residing in it.

Cache Initialization. Cache initialization refers to a special arrangement of blocks in the texture memory for a given orientation of reformatted plane(s). This arrangement maximizes performance. If the needed blocks can fit in the texture memory, i.e., if the number of blocks needed for cine MPR is less than the number of texture objects, cache initialization involves simply copying all the needed blocks into the texture memory.

More frequently, however, the needed blocks outnumber the texture objects, in which case some blocks are left out of the texture memory and need copying during rendering and interaction. A single texture object is reserved for this short-term caching and used as a temporary buffer for nonresident blocks. The goal of cache initialization is to minimize the load of copying blocks "on-the-fly" and also spread it approximately evenly among the frames, ensuring a regular frame rate. Moreover, by spreading the data transfer throughout the sequence, a higher frame rate, preferably higher than that required, is achieved. To implement this logic, cache initialization allocates an equal number of texture objects to each frame. The share of texture objects per frame, if not an integer, is rounded down. The few remaining texture objects are assigned one per frame until exhausted from the start of the sequence. When copying blocks into the assigned texture objects, the common blocks of a frame are given the priority. The remaining texture objects are filled with single-use blocks, any combination of which leads to the same performance.

A hypothetical numerical example should explain cache initialization further. Let's assume a sequence of 20 frames with 120 blocks per frame needed for cine MPR. Let's further assume that 2048 texture objects exist. Reserving one texture object for the short-term caching, the share of each frame is 2047/20 texture objects, or 102 upon rounding. There still remains (2048 – (102*20) –1) or 7 texture objects unassigned. These 7 texture objects are assigned one each to the first 7 frames in the sequence. So, eventually, the first 7 frames get 103 texture

objects each, whereas the share of the last 13 frames is 102. During visualization, 17 blocks per frame will be copied "on-the-fly" to render the first 7 frames and 18 to render the last 13 frames.

Cache Replacement. Whenever the user changes the orientation of the reformatted plane(s), a properly initialized cache (texture memory) no longer remains optimal. The distribution of blocks needs updating to match the changing viewing parameters. A possible solution is to perform cache initialization, as described above, for each intermediate orientation; however, that step has two problems. First, uploading the needed blocks (common or single-use) for the entire sequence is time-consuming and damaging to maintaining the necessary frame rate. Second, a thorough cache update optimized for the display of the entire sequence may be unnecessary if the orientation continues to change before the sequence is completed.

For cache replacement, we generate the list of needed blocks as before; however, we forego the common block determination during interaction. Comparing the existing texture memory occupancy against the new list of blocks produces two more lists: 1) the "stale-block" list or the list of resident blocks no longer needed, and 2) the "fresh-block" list or the list of nonresident blocks needed for visualization. Our cache replacement policy is to overwrite the blocks of the stale-block list with those in the fresh-block list. If the stale-block list is larger, all blocks from the fresh-block list are copied. If the fresh-block list is larger, the available texture objects holding "stale" blocks are split approximately equally among the frames. This process continues as long as the interaction continues. When the interaction is over, the cache is updated one more time to restore its contents to what they would be if the cache were initialized for the current orientation.

IV. EXTENSION OF THE ALGORITHM TO MULTIPLE DATA SETS

We described above how to perform cine MPR of a single 4-D image. Many practical applications require displaying two images together. This section explains how the ideas presented above are extended and applied to the visualization of multiple data sets.

A. Doppler Data Visualization

Although a Doppler image may be regarded as a single image, it is in fact the superimposition of two separate images: a Doppler blood flow image and an anatomical image. Typically, the color-coded Doppler flow information is displayed overlaid on the gray-level anatomical image. To switch Doppler image overlay on and off easily, and to manipulate the colormap dynamically, it is best to process the two images separately and combine them only at the very last stage for the final display on the computer screen. Our two-pass rendering approach helps achieve this. We describe below our rendering approach as well as the application of volume subdivision and caching in Doppler data visualization.

Volume Subdivision. The Doppler image typically occupies a portion of the volume the anatomical image occupies. Moreover, the spatial and temporal sampling rates (voxel size and frame rate, respectively) of the Doppler image may be different from those of the anatomical image. Disregarding such differences, the Doppler image is subdivided in exactly the same manner as the gray-level anatomical image. The block size, defined in terms of the number of voxels, is kept the same for both images to facilitate caching. If the voxel size differs, the physical dimensions of Doppler and anatomical image blocks will vary. The difference in physical dimensions, however, does not require altering any of the steps in rendering.

Rendering Reformatted Planes. As before, the reformatted plane is rendered as a mosaic of the texture-mapped polygons. Here, we repeat the process twice, once for the Doppler image and then for the anatomical image. Between the two passes, a stencil of the Doppler image is

created, which is used to prevent Doppler screen pixels from being overwritten by a gray-level anatomical image in the second pass. The color is applied to the Doppler image via a color look-up table (LUT), which is indexed with the interpolated Doppler flow values resulting from texture mapping (trilinear interpolation). The looked-up value is an RGB triplet, which is also the color applied to the screen pixel under consideration.

The steps remain the same for cine MPR visualization, except that a temporal alignment process must occur first if the two 4-D images do not have the same frame rate, i.e., the same number of frames. Temporal alignment equates the number of frames such that a Doppler image frame exists for each anatomical image frame and vice versa. For instance, if the Doppler image has half as many frames as the anatomical image, applying zero-order (nearest-neighbor) interpolation to the Doppler sequence along the time axis duplicates each Doppler frame. In reality, of course, no data are duplicated; instead, a table maintains the pointers of frames that form a pair. Nearest-neighbor interpolation is used even when one frame rate is not an integer multiple of the other. As a result, not every frame in the shorter sequence is duplicated with equal frequency.

Caching. The texture memory is partitioned between Doppler and anatomical images in proportion to image size. Since a texture object is the smallest unit of texture memory in our implementation, the texture objects are allocated to the two images in proportion of their 4-D size. Within each image's texture memory partition, cache initialization and cache replacement are performed exactly as described above. As before, a single texture object is reserved for short-term caching of nonresident blocks of both images.

B. Multimodality Visualization

An example of multimodality cardiac visualization is the fusion of real-time 3D ultrasound and gated cardiac single photon emission computed tomography (SPECT) images. This novel application we are developing combines complementary information: wall thickness and motion data from ultrasound and myocardial perfusion from SPECT. The fused images promise to make myocardial disease diagnosis more definitive.

The cine MPR problem in this case is similar to the Doppler data visualization, with a few minor differences. Like Doppler data visualization, the two data sets (real-time 3D ultrasound and gated SPECT) are subdivided. The texture memory allocation, as before, is proportional to their 4-D size. If the number of frames does not match, temporal interpolation is performed. A difference is that the two images are blended for the final display as opposed to the Doppler image hiding the anatomical image. The result, therefore, is the same, irrespective of the order in which the two images are rendered. One way to blend two images is to have the screen pixels alternate between the two images, vertically and horizontally. Consequently, the stencil buffer, in this case, resembles a checkerboard. As before, LUT is used to pseudocolor the SPECT image. The colormap represents the color scheme used clinically.

Image registration usually precedes multimodality fusion. Our group has developed 3-D image registration algorithms [15], which are applied to the current problem. Consequently, a 3-D transformation exists for each frame of the SPECT sequence, which aligns the SPECT frame with its corresponding ultrasound frame. This transformation is concatenated with any user-introduced transformation before cine MPR.

C. Single-Modality Visualization (3-D Stress Echocardiography)

3-D Stress echocardiography [16, 17], an emerging application of real-time 3-D ultrasound, compares the response of exercise or other forms of stress on the heart to the resting condition. This application requires that the pre- and post-stress sequences be displayed

alongside each other. The dynamic manipulation we provide enables a user to visualize any cross-section through the heart. The two views move in a coupled fashion during interaction. A common problem with stress echocardiography is the misalignment between pre- and post-stress images, which when displayed alongside may not represent the same anatomical cross-section. We use our aforementioned registration algorithm to correct for this misalignment.

Save for a few differences, the cine MPR of single-modality images is similar to multimodality fusion described above. The first difference is that the matching views from pre- and post-stress image sequences are presented in two separate side-by-side windows, instead of being fused in a single window. Moreover, the two sequences are temporally aligned for registration, but their display may or may not be so. When animated at the acquisition frame rate, pre- and post-stress reformatted views loop independently. Because the heart rate is significantly faster following stress, the heart in the post-stress image, quite correctly, beats more rapidly. The display can be temporally aligned if the user so wishes by slowing down the looping of the post-stress image. Volume subdivision, partitioning of the texture memory, caching and polygonal rendering are performed in an identical fashion.

V. RESULTS

We present, first of all, the effect of brick size on the performance (frame rate). Our first goal was to select the optimal block size for typical real-time 3-D ultrasound image sequences collected using a 2.5 MHz frequency real-time 3D scanner (Volumetrics, Inc., Durham, NC). Besides block size, the frame rate depends on several other factors: 4-D image size, image orientation, number of panels (reformatted planes), the panel size (viewport width and height in pixels), texture memory size, interaction, etc. The effects of the number of panels, panel size and texture memory size are not presented because they follow the expected trend. The higher the number of panels or the larger the panel size, the slower is the frame rate. Likewise, the greater the texture memory, the higher is the performance. The results reported are for three-panel display in all cases except the 3-D stress echocardiography application, which warrants only two panels. The panel size at 544 x 544 was kept as large as possible for three-panel display. The reported performance figures were obtained on a dual 2-GHz Pentium personal computer with Wildcat 6110 graphics (3DLabs, Milpitas, CA) and 2 GB of memory. The Wildcat 6110 graphics board is rated to have 128 MB of hardware-accelerated 3-D texture memory; however, given 2 bytes as the smallest texture element size, the effective texture memory was only 64 MB for the 8-bit ultrasound images used here. Accounting for additional overhead associated with allocating hundreds of texture objects, we empirically found the usable texture memory to be approximately 56 MB.

A. Performance Versus Block Size

The maximum achievable frame rates are shown in Fig. 4 for the three-panel cine MPR of a typical real-time 3-D ultrasound cardiac image consisting of 20 volumetric frames (128 x 128 x 512), collected at 25 fps. The voxel size was 1.1 x 1.1 x 0.27 mm. A snapshot of the three-panel cine MPR display is shown in Fig. 5. The frame rate, as mentioned above, depends on the orientation. Intuitively, the larger the cross-sectional area, the slower is the frame rate. We therefore report frame rate for both 0^0 and 45^0 orientations (Fig. 6), representing, respectively, the best and the worst cases computationally. Observed frame rates during interaction were also recorded and reported as a range because the frame rate depended on the orientation, which changed during interaction. From the plot, we concluded that 16 x 16 x 32 was the optimal block size because it helped achieve frame rates greater than 25 fps even during interaction; 16 x 32 x

32 and 32^3 are also acceptable block sizes. Everything else being equal, we observed that the preferred block size is the one that is "more cubic" in the physical space. Given 4:4:1 relative voxel dimensions, the 16 x 16 x 32 block size was more cubic than others. A perfectly cubic 16 x 16 x 64 block size was not optimal because its larger size relative to 16 x 16 x 32 offsets the benefits of a cubic block.

B. Performance Versus Image Size

We studied the performance of the cine MPR algorithm as a function of image size by varying the number of frames in a test 4-D image. Each frame was sized 256^3 (16 MB) and was uniformly sampled. In comparison, real-time 3-D ultrasound images had a higher sampling rate axially because of the inherently higher axial spatial resolution. The number of frames of the test data was varied from 4 to 40 in steps of 4, thus varying the 4-D image size from 64 MB to 640 MB. The block size was 16^3. As before, a higher maximum achievable frame rate was obtained in the 0^0 orientation for any given data size (Fig. 7). For either orientation, the initial flat portion of the plot refers to the situation in which the needed blocks fit completely in the available texture memory requiring no short-term caching. Performance decreases when texture memory usage exceeds the available memory, and this drop is roughly proportional to the texture memory supply and usage imbalance.

We also observe that the performance was significantly lower for a test image comparable in size to the 160-MB real-time 3-D ultrasound image described in the previous subsection. It is important to note that only about a third of the voxels in the real-time 3-D ultrasound data are valid image voxels because 3-D ultrasound is acquired on a 60^0 x 60^0 pyramid. The rest are background voxels, and blocks containing background voxels were excluded, reducing the data requirement. The test data, in comparison, were assumed to have no background voxels.

C. Cine MPR of Cardiac CT

The cine MPR algorithm presented here is general and applicable to non-ultrasound data, as well. To demonstrate generality, a reformatted plane of a 4-D cardiac CT image, acquired in a gated fashion on a Philips Mx8000 Quad scanner (Philips Medical Systems, Highland Heights, OH), is shown in Fig. 8. The dimensions of the 4-D image were 512 x 512 x 107 x 8 and the voxel size 0.5 x 0.5 x 2.4 mm. The desired frame rate was 10 fps, given the heart rate of 75 beats per minute. Fig. 9 shows the frame rate versus block size plot. Clearly, 32 x 32 x 16 is the optimal block size for this data set. Even when using the optimal block size, the frame rate falls slightly below the desired 10-fps frame rate in the 45^0 orientation and during interaction. We attribute this to the large data size (214 MB) in comparison to the limited texture memory (56 MB).

D. Doppler Data Visualization

We present results for a typical real-time 3-D Doppler image for a normally beating heart. The specific image used had 20 frames each of Doppler flow and anatomical data, frame size of 64 x 64 x 512 and 128 x 128 x 512, respectively, and 1.1 x 1.1 x 0.27 mm as the voxel size in both cases. One of the panels of the three-panel cine MPR display is shown in Fig. 10(a). The frame rates achieved were 40-95 fps for fixed orientations and 22-35 fps during interaction when using a block size of 16 x 16 x 32. The desired frame rate was 25 fps.

E. Multimodality Cardiac Image Fusion

In this application, real-time 3-D cardiac ultrasound images were registered with cardiac SPECT images. The temporally and spatially aligned images were fused and displayed in the

three-panel layout, a panel of which is shown in Fig. 10(b). In the example here, the real-time 3-D ultrasound image consisted of sixteen 128 x 128 x 512 frames (128 MB). The SPECT image had eight 64 x 64 x 27 frames (1 MB). Voxel sizes were 1.3 x 1.3 x 0.31 and 7.1 x 7.1 x 7.1 mm, respectively in the ultrasound and SPECT images. Since the real-time 3-D ultrasound data set was significantly larger, the block size (16 x 16 x 32) optimal for it was chosen for multimodality display. The frame rates achieved were 60-130 fps for fixed orientations and 30-35 fps during interaction. As before, the desired frame rate was 25 fps.

F. 3-D Stess Echocardiography

The expected performance in 3-D stress echocardiography was obtained with a pair of typical 4-D images. The pre-stress image had 20 frames, whereas the post-stress image had 12. Each frame had 128 x 128 x 512 voxels, and voxel dimensions were 1.1 x 1.1 x 0.27 mm. Using the optimal 16 x 16 x 32 brick size determined above, the maximum frame rates achieved were 40-52 fps, including those during interaction for a two-panel display. Fig. 11 has the usual two-panel display, where the left panel shows the pre-stress image and the right panel, the spatially aligned post-stress image.

VI. DISCUSSION

We have presented hardware-accelerated interactive MPR of dynamic volumetric cardiac images. Our method permits one to visualize a number of planes through a 4-D image simultaneously and manipulate them freely without any perceptible delay. Real-time 3-D ultrasound, the latest advance in 4-D ultrasound imaging, combined with the cine MPR algorithms we have developed, heralds a new era in the visualization of cardiac anatomy and motion. No longer will the image collection be partial, nor will there be any limits on exploring the resulting 4-D image through our powerful and flexible volumetric image exploration tool. When used in conjunction with telemedicine, it is with this tool that a clinician will virtually re-scan the patient's heart.

Interactive MPR of dynamic 3-D data is computationally and memory-access intensive, and achieving the needed performance is difficult without hardware acceleration. Developing custom hardware for this application is not uncommon [18]. A key advantage of our implementation is that it uses commercially available graphics boards. It is also independent of a specific hardware and will function in conjunction with any graphics board supporting 3-D texture mapping and OpenGL. Although significantly less costly than custom hardware, the 3-D texture mapping hardware may still be relatively expensive presently. However, most computing hardware become affordable and powerful with time.

We regard using volume subdivision and caching in novel ways as our main contribution. 2-D/3-D Image subdivision techniques, especially the familiar special cases, quadtree and octree decompositions [19], are common in image processing and computer graphics. Volume subdivision in the context of even texture mapping has been reported; however, the optimal block size has been kept equal to the available texture memory or the on-chip cache [20]. Another approach involving smaller blocks has used temporal correlation to discard identical blocks across frames [13]. Our approach of subdividing a volume into extremely fine blocks and using the blocks as a means to separate desired data from undesired is novel. The need to manage a number of small data blocks in the texture memory has quite naturally led to the subject of caching. New cache initialization and replacement strategies needed to be developed to optimize the performance. The existing least recently used (LRU) and least frequently used (LFU) cache replacement policies [21] were not effective because the cache entities (image blocks) were

needed at regular and equal frequency, whereas these policies rely on arbitrary and unequal demand for the cache entities for their effectiveness. Since the cache entities were identical in size, strategies based on size differences [21] could not be used, either.

The benefits of volume subdivision and caching are obvious. 4-D Cardiac images, as large as four times the effective hardware-accelerated texture memory (56MB) could be visualized at interactive rates without altering the underlying heart motion. Volume subdivision and caching, as presented by us, also provide a general framework for more complex visualization involving most 3-D/4-D medical images. Visualization of multiple data sets, as in Doppler as well as multi- and single-modality data visualization, was an example showing the power of this framework. We are utilizing this framework also for dynamic volume rendering, which is made faster by reduced data requirement resulting from discarding any blocks either containing transparent voxels or sculpted out for cutaway views. Caching is helping to optimize rendering speed further.

The use of accelerated texture memory, the strength of our approach, may also be a weakness. Using hardware acceleration, we indeed obtain frame rates that an equivalent CPU-only implementation cannot match. However, the amount and speed of the texture memory also set an upper limit on performance, surpassing which requires more hardware resources. Fortunately, hardware capabilities grow with time, and our implementation will immediately benefit from such advances. For additional performance, using a graphics cluster may be a solution as suggested earlier [22].

Further algorithmic enhancements may improve the performance of our implementation during interaction. Since the orientation changes gradually during interaction, algorithms can be developed that extrapolate user interaction history to predict the expected future orientation of the reformatted planes. A more advanced cache replacement strategy will make use of this information while updating the cache. Additionally, when visualizing two images, defining a hard partition in the texture memory is slightly suboptimal. An enhancement may be to divide texture objects between the two images adaptively as dictated by the changing block requirements.

In conclusion, we have provided a fast, interactive method for cine MPR of time-varying volumetric images, which is based on our novel use of volume subdivision and caching concepts, and the use of commercially available 3-D texture mapping hardware. Our approach further provides a general-purpose framework for visualizing 3-D/4-D medical images of any type, modality and organ in many different ways.

APPENDIX

CALCULATION OF POLYGON VERTICES AND TEXTURE COORDINATES

The following assumes that the camera is always fixed and that the reformatted plane is the intersection of the $z = 0$ plane with the image volume. To change the orientation of the reformatted plane, the image volume is transformed. The simpler case of undivided volume is considered first.

A 3-D texture is a unit cube with r, s and t coordinates ranging in [0,1]. We assume that the image volume sits in the positive octant and that one of its vertices coincides with the origin of the world coordinate system. We further assume that $\boldsymbol{T_s}$, a scaling matrix, defines the transformation between the unit texture cube and the image volume. Pre-multiplying the vertices of the texture cube with $\boldsymbol{T_s}$ results in the vertices of the image volume. A one-to-one matching between the vertices of the texture cube and the image volume is thus established.

We assume next that the image volume is transformed in the world coordinates by the modelview matrix $\boldsymbol{M}$. To achieve one-to-one matching as before, we now need to pre-multiply the vertices of texture cube by $\boldsymbol{MT_s}$. To perform MPR in the current orientation, the intersection of each of the 12 sides of the image volume with the $z = 0$ plane is tested. If $\{v_1, v_2, v_3\}$ were the intersection points ordered counter-clockwise, the reformatted plane is obtained by rendering the polygons $\{v_1, v_2, v_3\}$. Corresponding texture coordinates are obtained by transforming $\{v_1, v_2, v_3\}$ back to the texture coordinate system, i.e., multiplying polygon vertices by $(\boldsymbol{MT_s})^{-1}$.

When the image volume is subdivided, an array of matrices $\boldsymbol{R_1}, \boldsymbol{R_2}, \ldots \boldsymbol{R_N}$ exists, which defines the transformation of the image volume to blocks 1 to N, respectively. The intersection of each block with the $z = 0$ plane is searched. If the i^{th} block is found to have intersections, pre-multiplying the polygon vertices with $(\boldsymbol{MR_iT_s})^{-1}$ yields the corresponding texture coordinates.

ACKNOWLEDGMENTS

The authors acknowledge Department of Cardiovascular Medicine of The Cleveland Clinic Foundation for providing the real-time 3-D ultrasound images and Philips Medical Systems for cardiac CT image.

REFERENCES

[1] J. A. Panza, "Real-time three-dimensional echocardiography: An overview," *International Journal of Cardiac Imaging*, vol. 17, pp. 227-235, 2001.

[2] M. Ahmad, "Real-time three-dimensional echocardiography in assessment of heart disease," *Echocardiography*, vol. 18, pp. 73-77, 2001.

[3] C. R. Hazard and G. R. Lockwood, "Theoretical assessment of a synthetic aperture beamformer for real-time 3-D imaging," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 46, pp. 972-980, 1999.

[4] M. Halliwell, H. Key, D. Jenkins, P. C. Jackson, and P. N. Wells, "New scans from old: digital reformatting of ultrasound images," *British Journal of Radiology*, vol. 62, pp. 824-829, 1989.

[5] N. M. Hylton, W. S. Chung, E. H. Botvinick, N. B. Schiller, P. Sheldon, and L. Kaufman, "Oblique reformatting of multislice magnetic resonance images for improved visualization of coronary arteries," *Journal of Digital Imaging*, vol. 3, pp. 34-37, 1990.

[6] K. A. Bradshaw, D. Pagano, R. S. Bonser, I. McCafferty, and P. J. Guest, "Multiplanar reformatting and three-dimensional reconstruction: for pre-operative assessment of the thoracic aorta by computed tomography," *Clinical Radiology*, vol. 53, pp. 198-202, 1998.

[7] A. Kaufman, *Volume visualization*. Washington: IEEE Computer Society Press, 1991.

[8] B. Cabral, N. Cam, and J. Foran, "Accelerated volume rendering of tomographic reconstruction using texture mapping hardware," *Workshop on Volume Visualization*, 1994, pp. 91-98.

[9] R. Westermann and T. Ertl, "Efficiently using graphics hardware in volume rendering applications," *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, 1998, pp. 169-176.

[10] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner, and T. Ertl, "Interactive volume rendering on standard PC graphics hardware using multi-textures and multi-stage rasterization," *Proceedings of the SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware*, 2000, pp.109-118.

[11] R. Z. Freidlin, C. J. Ohazama, A. E. Arai, D. P. McGarry, J. A. Panza, and B. L. Trus, "NIHmagic: 3D visualization, registration and segmentation tool," *Proceedings of SPIE*, vol. 3905, pp. 194-201, 2000.

[12] E. B. Lum, K.-L. Ma, and J. Clyne, "Texture hardware assisted rendering of time-varying volume data," *Proceedings of the IEEE Visualization Conference*, 2001, pp. 263-270.

[13] J. Hwang, J.-S. Kim, I.-Y. Kim, and S. I. Kim, "Real-time volume rendering of 4-D image using 3-D texture mapping," *Proceedings of SPIE*, vol. 4319, pp. 557-563, 2001.

[14] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer graphics: principles and practice in C*, 2nd ed. Reading, Mass.: Addison-Wesley, 1996.

[15] R. Shekhar and V. Zagrodsky, "Mutual information-based rigid and nonrigid registration of ultrasound volumes," *IEEE Transactions on Medical Imaging*, vol. 21, pp. 9-22, 2002.

[16] M. Ahmad, T. Xie, M. McCulloch, G. Abreo, and M. Runge, "Real-time three-dimensional dobutamine stress echocardiography in assessment stress echocardiography in assessment of ischemia: comparison with two-dimensional dobutamine stress echocardiography," *Journal of the American College of Cardiology*, vol. 37, pp. 1303-1309, 2001.

[17] R. Shekhar, V. Zagrodsky, M. Garcia, and J. D. Thomas, "3D Stress Echocardiography: A novel application based on registration of real-time 3D ultrasound images," *Proceedings of CARS,* 2002, pp. 873-878.

[18] L. Capineri, L. Masotti, S. Rocchi, F. Andreuccetti, M. Cerofolini, and A. Tondini, "Nearly real-time visualization of arbitrary two-dimensional sections from three-dimensional acquisition," *Ultrasound in Medicine & Biology*, vol. 22, pp. 319-328, 1996.

[19] M.-M. Yau and S. N. Srihari, "Hierarchical data structure for multidimensional digital images," *Communications of the ACM*, vol. 26, pp. 504-515, 1983.

[20] B. Lichtenbelt, S. Naqvi, and T. Malzbender, "Volumetric data organization method that allows for cache efficient rendering speedups and efficient graphics hardware design," *US Patent 6,144,383*, November 7, 2000.

[21] J. Dilley and M. Arlitt, "Improving proxy cache performance: Analysis of three replacement policies," *IEEE Internet Computing*, vol. 3, pp. 44-50, 1999.

[22] J. Kniss, P. McCormick, A. McPherson, J. Ahrens, J. Painter, A. Keahey, and C. Hansen, "Interactive texture-based volume rendering for large data sets," *IEEE Computer Graphics and Applications*, vol. 21, pp. 52-61 2001.

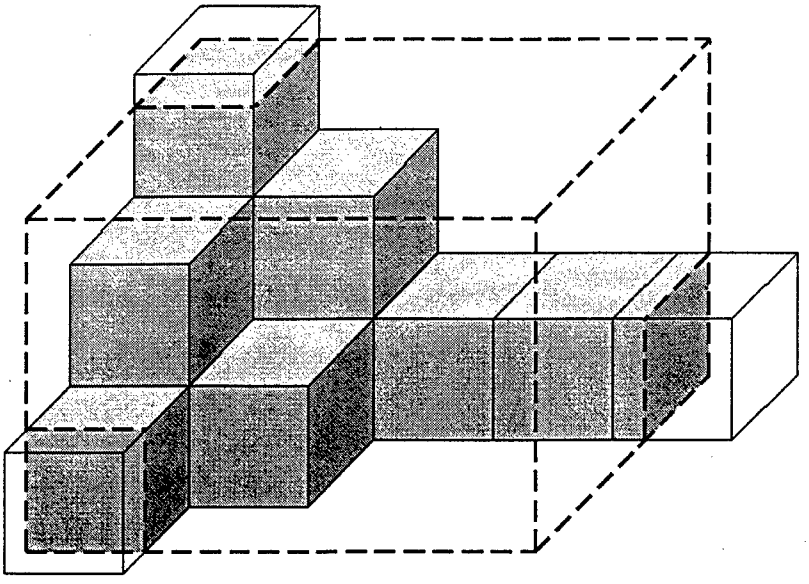


Fig. 1. A schematic showing volume subdivision. The large dashed box represents the original image volume, and the smaller gray boxes are the image blocks. Note the partially filled image blocks on the edges of the image volume.

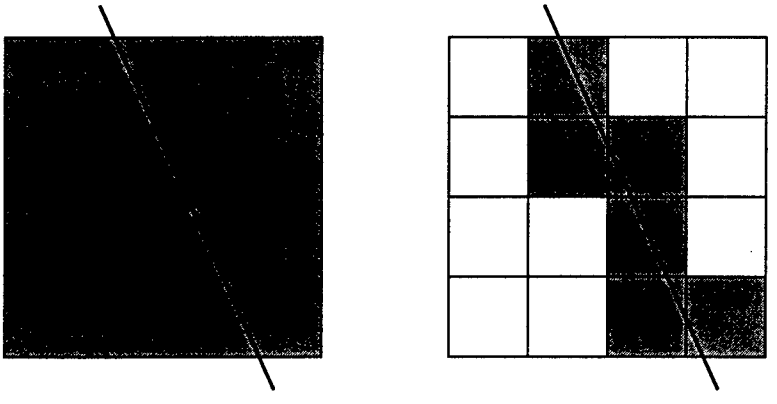


Fig. 2. A 2-D schematic showing data reduction following volume subdivision. If the oblique line were the reformatted plane, only shaded blocks would be needed for visualization.

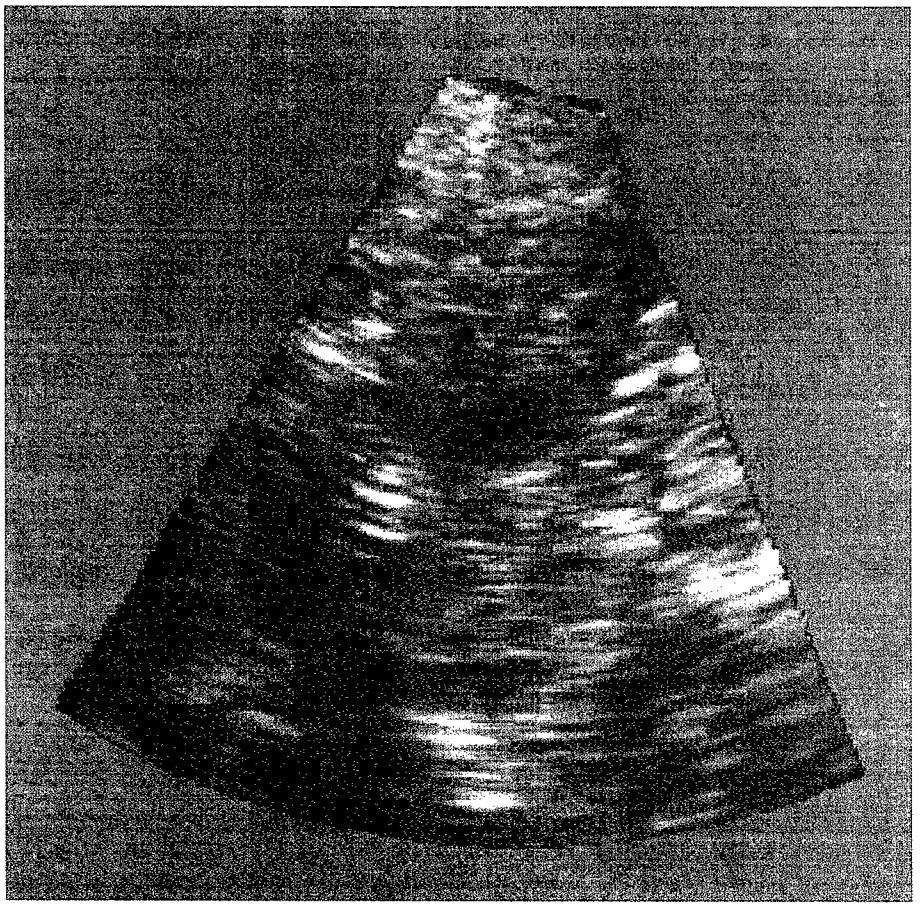


Fig. 3. A reformatted plane illustrating seamless tiling of the polygons and the underlying mosaic by intentionally coloring the blocks with randomly selected colors.

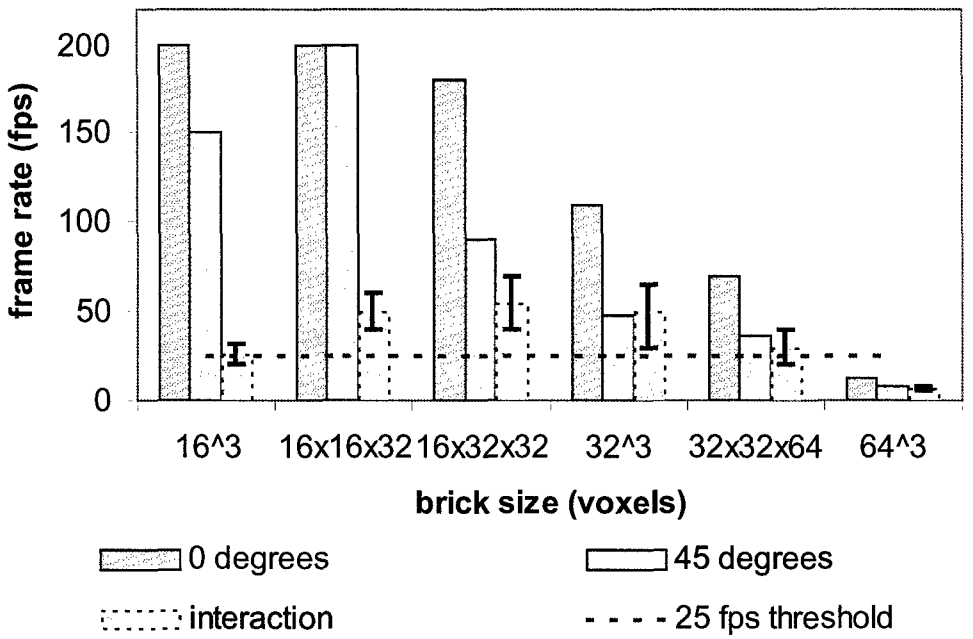


Fig. 4. Maximum achievable frame rate during three-panel cine MPR as a function of the block size for a typical real-time 3-D ultrasound cardiac image sequence for 0^0 and 45^0 orientations and during interaction. Several block sizes ensure greater than 25 fps performance. The optimal block size of 16 x 16 x 32 voxels provides the best performance.

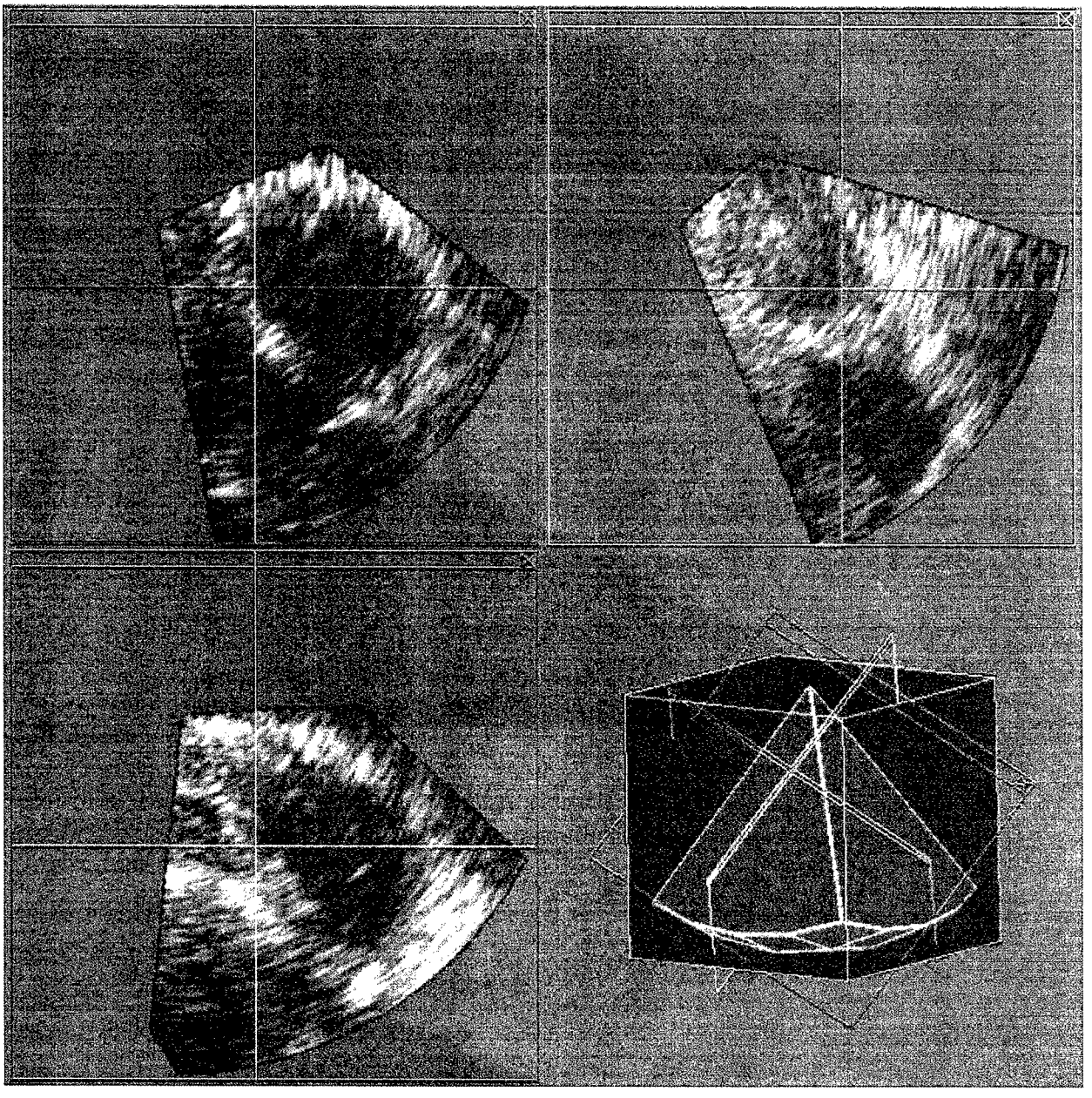
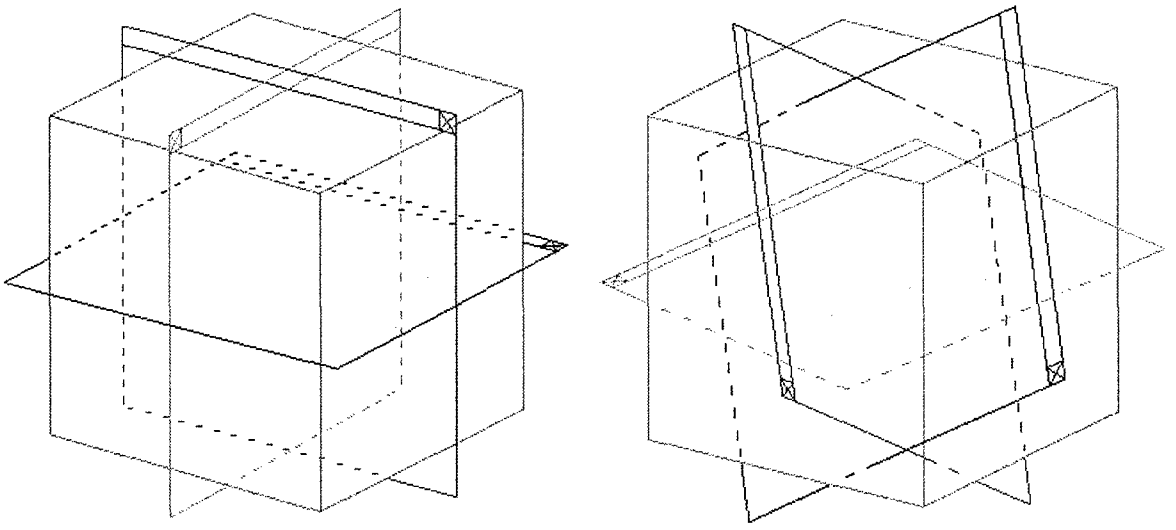


Fig. 5. A snapshot of three-panel cine MPR of real-time 3-D cardiac image sequence. The lower-right panel is a reference schematic that shows the orientation of the three reformatted planes with respect to the original image volume in a color-coded fashion.



0^0 Orientation 45^0 Orientation

Fig. 6. Least (0^0) and most (45^0) computationally demanding orientations of reformatted planes.

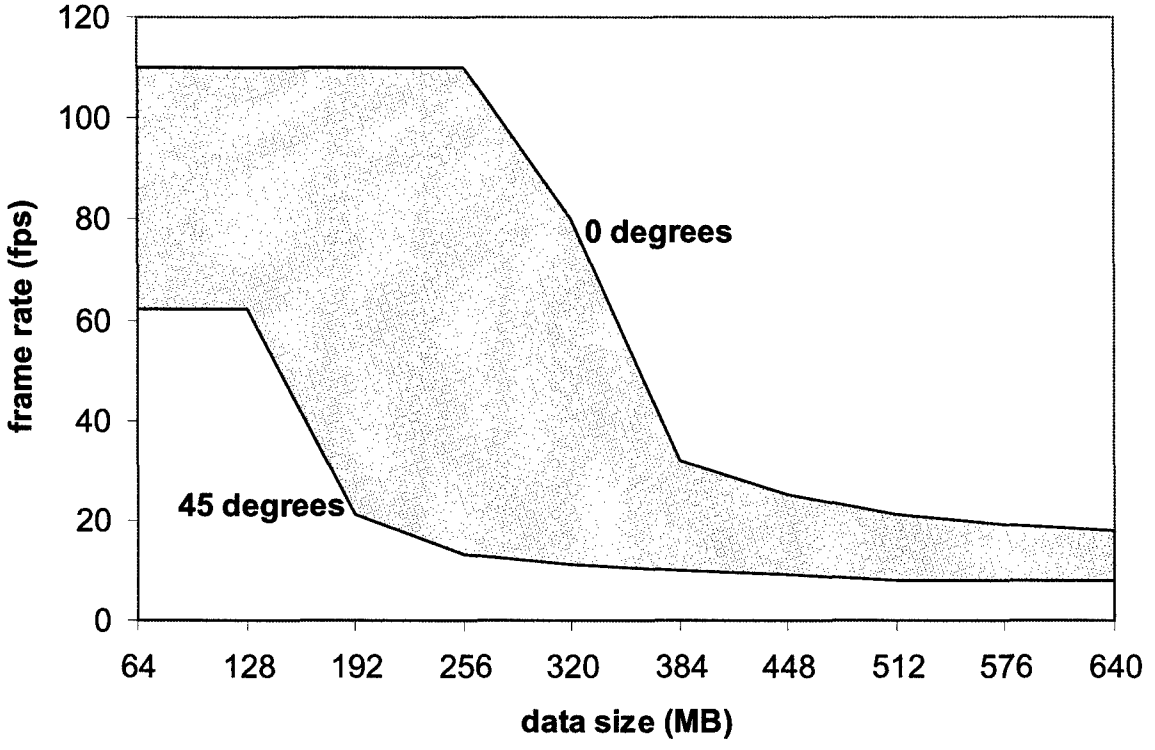


Fig. 7. Maximum achievable frame rate during three-panel cine MPR versus 4-D image size. The 0^0 and 45^0 orientations define the upper and the lower limits of performance, respectively. The initial flat region represents the case in which the needed image data fit entirely in the available texture memory.

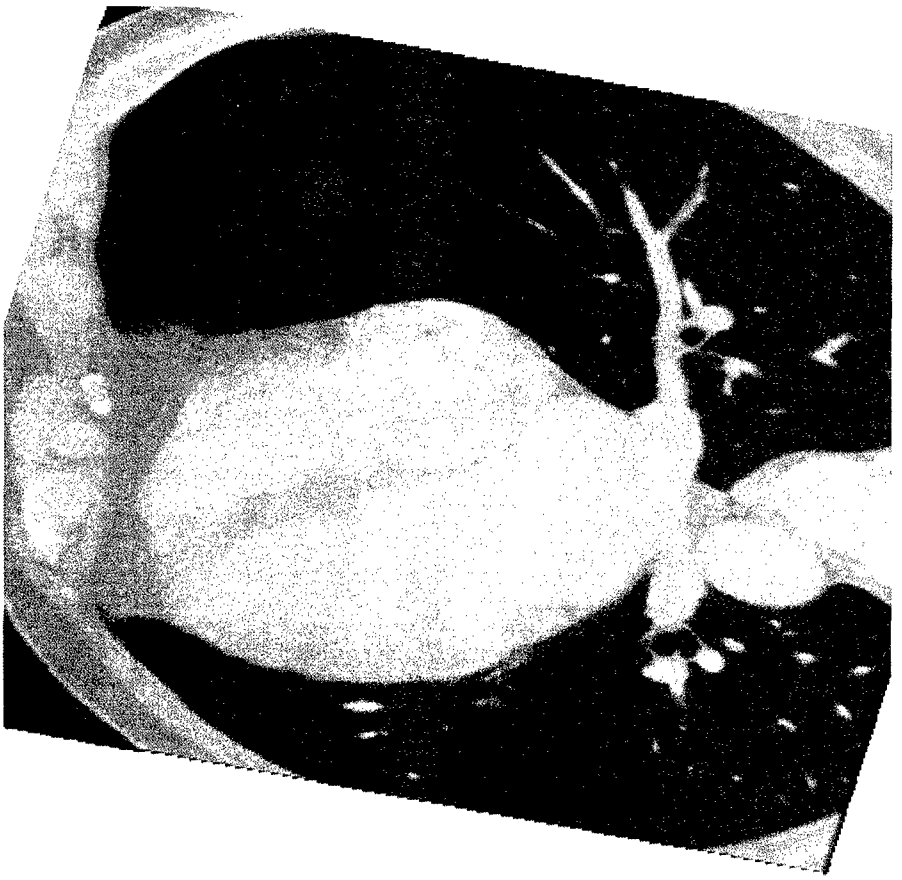


Fig. 8. A reformatted plane through the cardiac CT image.

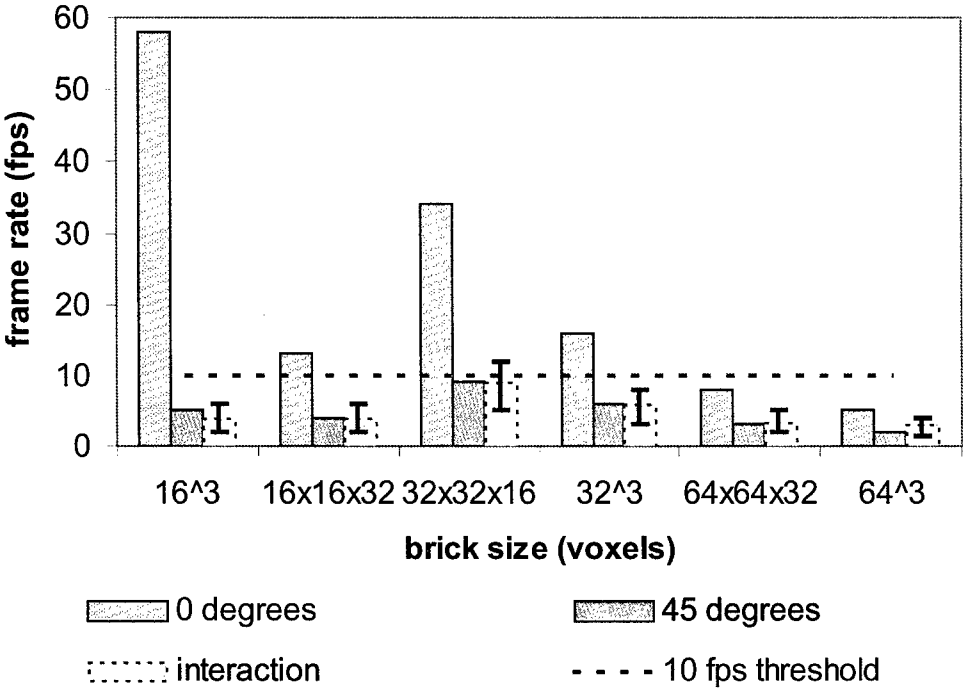
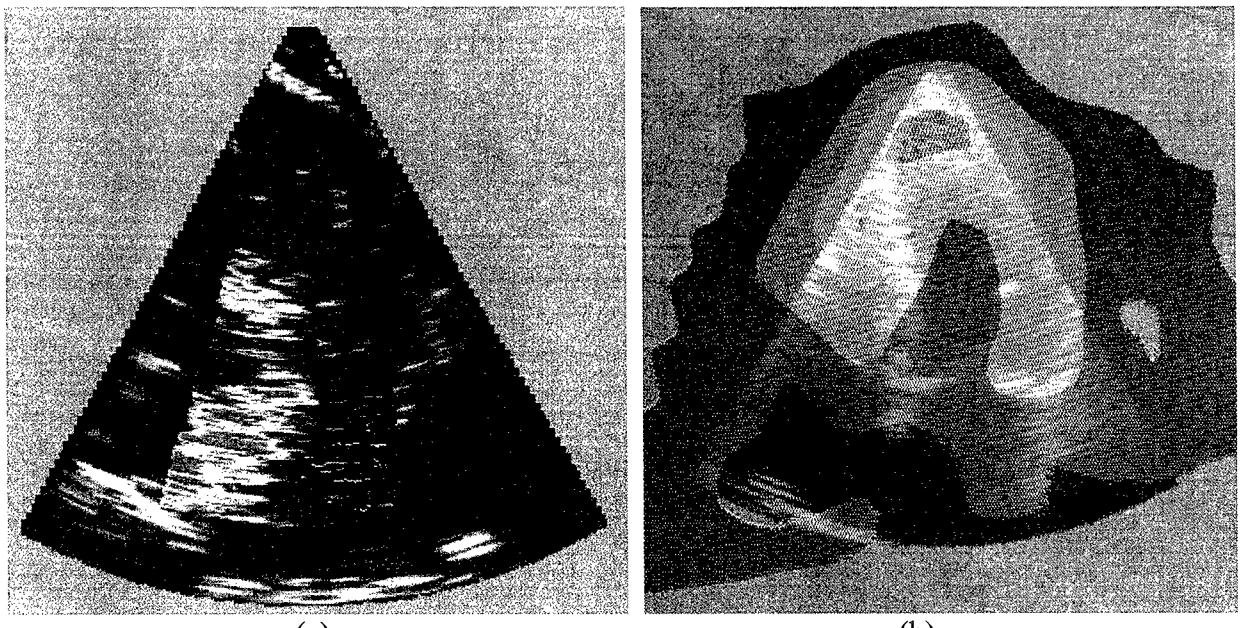


Fig. 9. Maximum achievable frame rate during three-panel cine MPR as a function of the block size for a typical 4-D cardiac CT image for 0^0 and 45^0 orientations and during interaction. The optimal block size is 32 x 32 x 16 voxels.



(a) (b)

Fig. 10. A reformatted plane through the Doppler ultrasound image (a) and fused ultrasound + SPECT image (b).

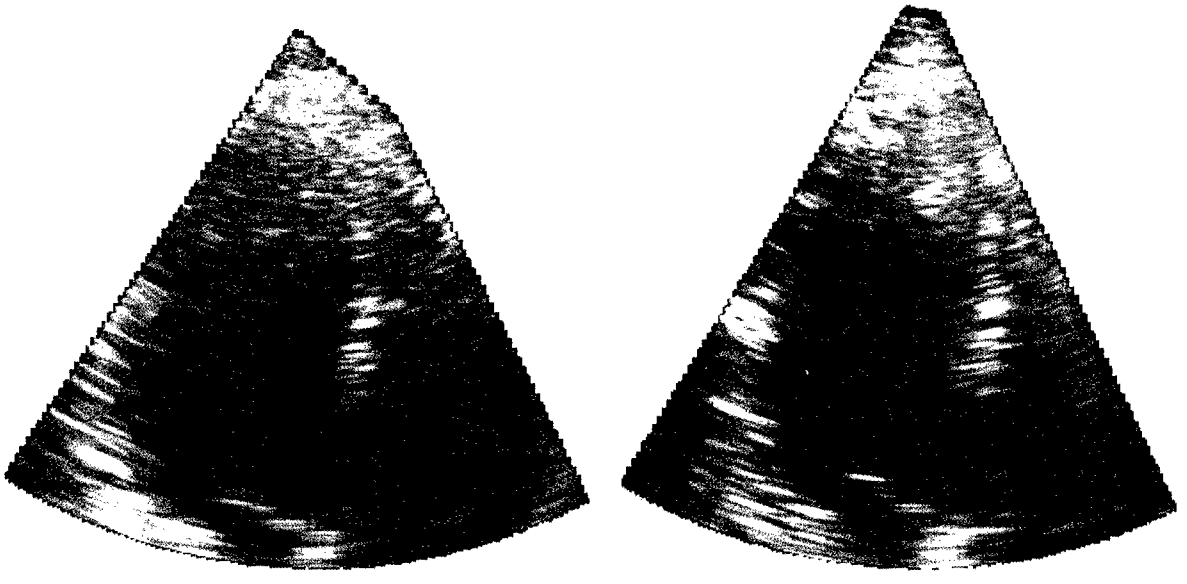


Fig. 11. A cross-section through registered pre- and post-stress images used in the 3-D stress echocardiography application.

DEPARTMENT OF THE ARMY
US ARMY MEDICAL RESEARCH AND MATERIEL COMMAND
504 SCOTT STREET
FORT DETRICK, MARYLAND 21702-5012

REPLY TO
ATTENTION OF:

MCMR-RMI-S (70-1y) 28 July 03

MEMORANDUM FOR Administrator, Defense Technical Information Center (DTIC-OCA), 8725 John J. Kingman Road, Fort Belvoir, VA 22060-6218

SUBJECT: Request Change in Distribution Statement

1. The U.S. Army Medical Research and Materiel Command has reexamined the need for the limitation assigned to technical reports written for this Command. Request the limited distribution statement for the enclosed accession numbers be changed to "Approved for public release; distribution unlimited." These reports should be released to the National Technical Information Service.

2. Point of contact for this request is Ms. Kristin Morrow at DSN 343-7327 or by e-mail at Kristin.Morrow@det.amedd.army.mil.

FOR THE COMMANDER:

Encl

PHYLIS M. RINEHART
Deputy Chief of Staff for
Information Management

ADB233865
ADB265530
ADB244706
ADB285843
ADB240902
ADB264038
ADB285885
ADB274458
ADB285735
ADB286597
ADB285707
ADB274521
ADB259955
ADB274793
ADB285914
ADB260288
ADB254419
ADB282347
ADB286860
ADB262052
ADB286348
ADB264839
ADB275123
ADB286590
ADB264002
ADB281670
ADB281622
ADB263720
ADB285876
ADB262660
ADB282191
ADB283518
ADB285797
ADB269339
ADB264584
ADB282777
ADB286185
ADB262261
ADB282896
ADB286247
ADB286127
ADB274629
ADB284370
ADB264652
ADB281790
ADB286578
ADB264750
ADB282776
ADB286264
ADB260563
ADB277918
ADB286365
ADB275327
ADB286736
ADB286137
ADB286146
ADB286100
ADB286266
ADB286308
ADB285832